SIEMENS

SIMATIC

STEP 7 V5.1 编程

使用手册

重要提示，目录

安全指南

本手册包括应该遵守的注意事项，以保证你自己的生命安全，保护产品和所连接的设备。这些注意事项在本手册中采用警示三角形加以突出强调，并根据危险等级注明如下：

危险（Danger）
表示若不采取适当的预防措施，将造成死亡、严重的人身伤害或重大的财产损失。

警告（Warning）
表示若不采取适当的预防措施，将可能造成死亡、严重的人身伤害或重大的财产损失。

小心（Caution）
表示若不采取适当的预防措施，将可能造成轻微的人身伤害或财产损失。

注意（Note）
提醒你对与产品有关的重要信息、产品的处置或文件的特别部分，应格外注意。

合格人员

只有合格人员才允许安装和操作这一设备。合格人员规定为根据既定的安全惯例和标准批准进行试运行、接地和为电路、设备和系统加装标签的人员。

正确使用

注意如下：

警告
本装置及其组件只能用于产品目录或技术说明书中阐述的应用，并且只能与西门子公司认可或推荐的其它生产厂的装置或组件相连接。
本产品只有在正确的运输、贮存、组装和安装的情况下，按建议方式进行运行和维护，才能正确而安全地发挥其功能。

商标

SIMATIC®、SIMATIC HMI®和SIMATIC NET®为西门子公司的注册商标。
本手册中所及其它名称也可能是注册商标，禁止未经允许为第三方所使用。

西门子股份有限公司
自动化与驱动集团
工业自动化系统部

郑重声明
我们已核对过，本手册的内容与所述硬件和软件相符。但错误在所难免，不能保证完全的一致。本手册中的内容将定期审查，并在下一版中进行修正。欢迎提出改进意见。

邮政信箱4848，纽伦堡D- 90327

重要提示

目的

本手册详细阐述了STEP 7进行编程，为安装和调试软件提供支持。本手册解释了如何生成程序，并对用户程序组件作了说明。

本手册的使用对象是那些使用STEP 7和SIMATIC S7自控系统实现控制任务的人员。

我们建议你通过手册《STEP 7 V5.1使用入门》中的例子，来了解STEP 7。这些例子对 “ 使用STEP 7编程 ” 的主题作了简单的介绍。

所需基本知识

为了很好理解本手册，需要具有自动化技术的一般知识。

另外，还应熟悉安装有Windows 95/98/2000或Windows NT操作系统的计算机或PC一类的工具的使用（例如编程器等）。

手册的应用范围

本手册适用于STEP 7编程软件的5.1版。

在线帮助

集成在软件中的在线帮助是本手册的补充。在线帮助的目的是为你提供详细的软件使用帮助。

帮助系统通过多个界面集成在软件中：

- 在Help菜单中有多个菜单命令可以选择：使用“Contents（内容）”命令，可以打开Step 7的帮助索引。
- Using Help（使用帮助）提供有详细的在线帮助使用说明。
- 上下文相关帮助可以提供关于当前的文本信息，例如，一个打开的对话框或一个激活的窗口。你可以通过点击“Help”按钮或按动F1，打开文本相关的帮助。
- 状态栏提供有其它形式的上下文相关帮助。当鼠标放在某个菜单命令上时，它为每个菜单命令显示一个简短的解释。
- 当鼠标短时放在一个工具栏的图标上时，也能为每个图标显示一个简短的解释。

如果你更愿意阅读打印出来的在线帮助，你可以打印每个帮助主题、工作簿或整个在线帮助。

本手册是从基于HTML的STEP 7帮助中摘取出来的。详细过程请参阅STEP 7帮助。由于手册和在线帮助的结构几乎一样，所以能够很容易地在手册和在线帮助之间进行转换。

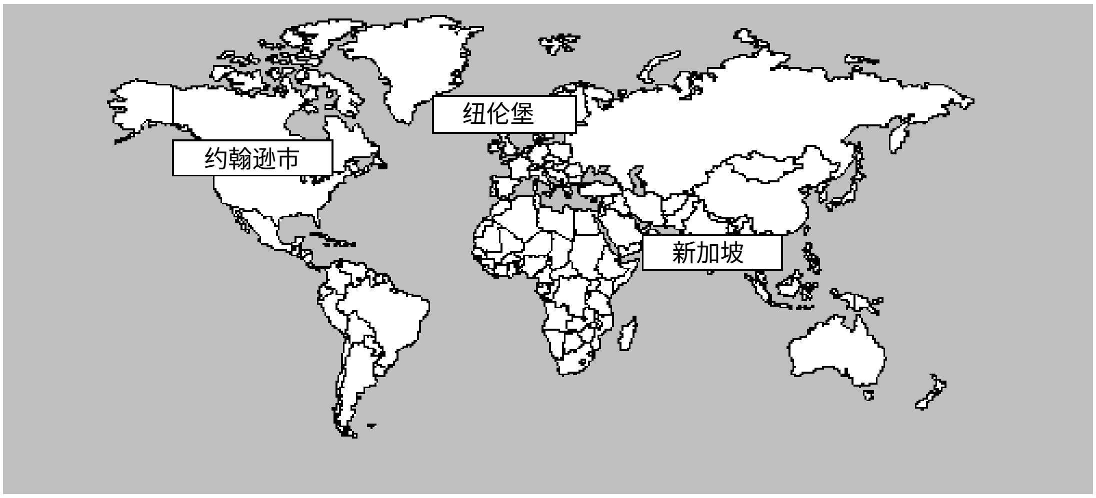
关于资料的反馈意见

我们需要您的支持，以帮助我们为您及将来的STEP 7用户提供尽可能好的资料。 如果关于本手册或在线帮助，您有任何意见或建议，请填写本手册最后的调查表，并按所述地址寄与我们。并请附上您个人对这些资料的评述。

SIMATIC培训中心

西门子公司还提供有许多培训课程，介绍SIMATIC S7自动化系统。详情请与您所在地区的培训中心联系，或与德国纽伦堡（邮编D-90327）的总部培训中心联系：
电话： +8610 64721888
电话： +49 （911） 895-3200

SIMATIC客户支持热线



昼夜值班，遍布全球：

面向全球（纽伦堡） 技术支持
（自由联系）
当地时间：星期一—星期五7:00 至17:00
电话：+49 （180） 5050 222
传真：+49 （180） 5050 223
E-Mail：techsupport@ ad.siemens.de
GMT：+1:00

面向全球（纽伦堡） 技术支持
（自由联系，但只适用于SIMATIC卡）
当地时间：星期一—星期五00:00至24:00
电话：+49 （911） 895-7777
传真：+49 （911） 895-7001
GMT：+01:00

欧洲/非洲
（纽伦堡） 授权
当地时间：星期一—星期五
7:00 至17:00
电话：+49 （911） 895-7200
传真：+49 （911） 895-7201
E-Mail：authorization@
nbgm.siemens.de
GMT：+1:00

美国（约翰逊市）
技术支持和授权
当地时间：星期一—星期五
08:00:00 至19:00:00
电话：+1 423 461-2522
传真：+1 423 461-2289
E-Mail：simatic.hotline@
sea.siemens.com
GMT：-5:00

亚洲/澳在利亚（新加坡）
技术支持和授权
当地时间： 星期一—星期五
8:30至17:30
电话：+65 740-7000
传真：65 740-7001
E-Mail：simatic.hotline@
sae.siemens.com.sg
GMT：+8:00

SIMATIC热线所使用的语言一般为德语和英语，另外对于授权热线，还使用法语、意大利语和西班牙语。

SIMATIC客户支持在线服务

SIMATIC客户服务支持部门，通过其在线服务，还可为您提供与更丰富的有关SIMATIC产品的其它信息：

- 从Internet在
  - 网址http://www.ad.siemens.de/simatic上，可以获得一般的当前信息。
- 通过下述方式，可以获得当前产品信息的说明和有用的下载资料：
  - 从Internet在网址http://www.ad.siemens.de/simatic-cs上
  - 通过号码为+49（911）895-7100的纽伦堡公告牌系统（BBS）（SIMATIC客户支持邮箱）。

要访问邮箱，应使用速度至少为V.34（28.8Kbps）的调制解调器，并按以下参数设置：8，N，1，ANSI，或通过ISDN（x.75, 64 Kbps）拨号。

- 在我们的客户服务代表处数据库中，您可以找到您当地的自动化与驱动集团客户服务代表处：
  - 从Internet网址：
  - http://www3.ad.siemens.de/partner/search.asp?lang=en
  - http://www.ad.siemens.com.cn/about_us/contact.asp

目录

1 产品入门

1.1 STEP 7 概述

何谓STEP 7

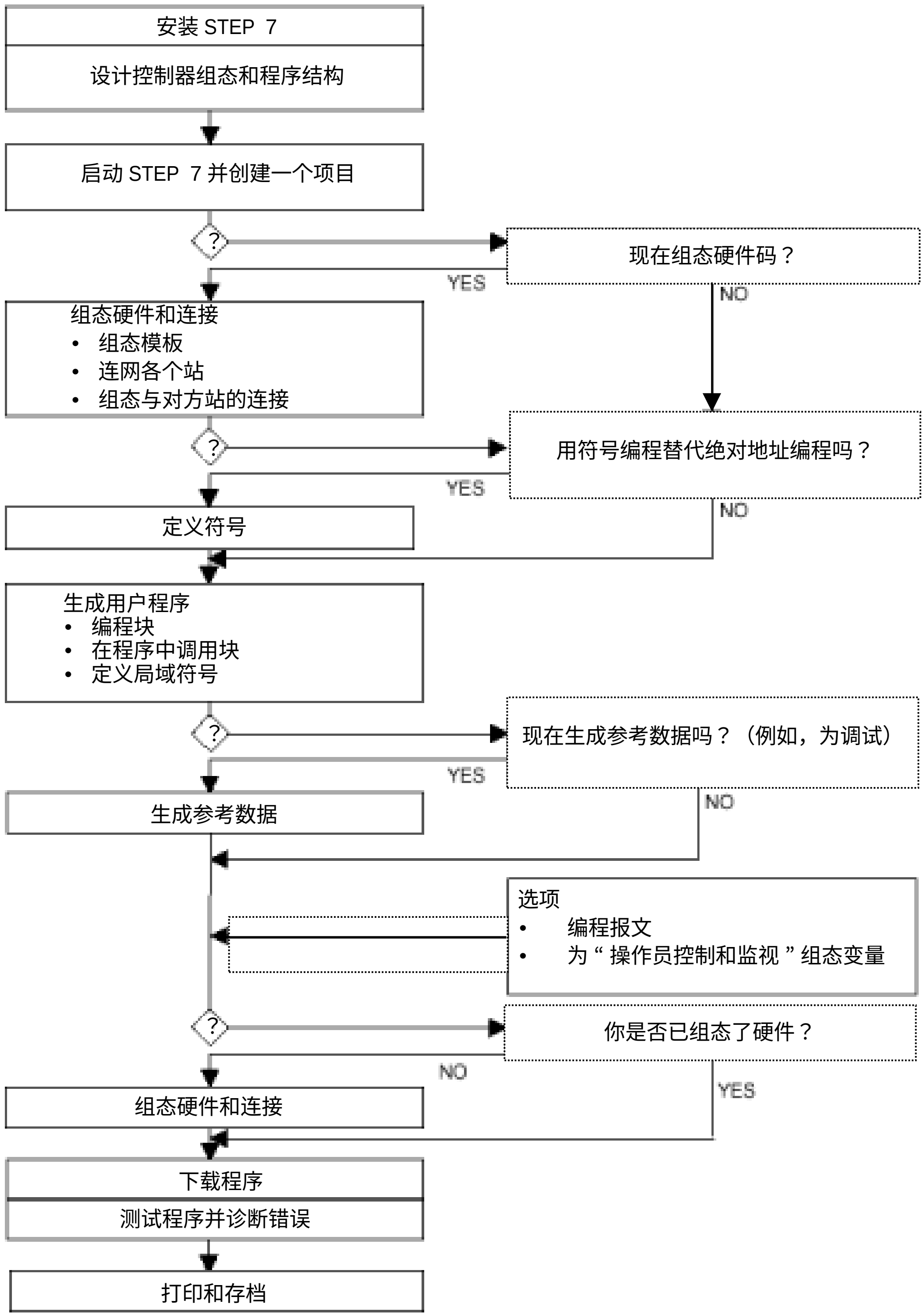
STEP 7 是用于 SIMATIC 可编程逻辑控制器组态和编程的标准软件包。它是SIMATIC 工业软件的组成部分。有下列版本的 STEP 7 标准软件包：

- 用于SIMATIC S7-200上简单单站应用的STEP 7 Micro/DOS和STEP 7 Micro/WIN。
- 用于使用带有各种功能的SIMATIC S7-300/ST-400、SIMATICM7-300/M7-400和SIMATIC C 7的STEP 7：
  - 可通过选择SIMATIC工业软件中的软件产品进行扩展（见“ STEP 7标准软件包的扩展应用 ”一节）
  - 为功能模板和通讯处理器赋值参数
  - 强制和多处理器模式
  - 全局数据通讯
  - 使用通讯功能块的事件驱动数据传送
  - 组态连接

STEP 7是本用户手册讨论的主题，STEP 7 Micro在《STEP 7 Micro/DOS用户手册》中描述。

基本任务

当你用 STEP 7 创建一个自动化解决方案时，有一系列的基本任务。下图所示为大多数项目需要执行的任务，并把这些任务分配到基本程序中。它会提示你与之相关的章节，因此你可以很方便地在整个手册中查找与任务相关的信息。


安装 STEP 7
设计控制器组态和程序结构
启动 STEP 7 并创建一个项目
?
现在组态硬件码？
YES
NO
组态硬件和连接
• 组态模板
• 连网各个站
• 组态与对方站的连接
?
用符号编程替代绝对地址编程吗？
YES
NO
定义符号
生成用户程序
• 编程块
• 在程序中调用块
• 定义局域符号
?
现在生成参考数据吗？（例如，为调试）
YES
NO
生成参考数据
选项
• 编程报文
• 为“操作员控制和监视”组态变量
?
你是否已组态了硬件？
NO
YES
组态硬件和连接
下载程序
测试程序并诊断错误
打印和存档

可选步骤

如上图所示，你有两个可选步骤：

- 先组态硬件然后编程块。
- 也可以先编程块而不组态硬件。这种方法建议用于维修和维护工作，例如，集成编程的块到一个已有的项目中。

各个步骤的简要说明

- 安装及授权
  第一次使用STEP 7时，要进行安装并将授权从磁盘传至硬盘（见“ 安装STEP 7 ”和“ 授权 ”一节）。
- 设计你的控制器
  在使用STEP 7之前，设计你的自动化解决方案。将过程分割为单个的任务以生成一个组态图表（见“ 设计一个自动化项目的基本步骤 ”一节）。
- 设计程序结构
  使用STEP 7中可用的块将你的控制器设计草案中所描述的任务转化为程序结构（见“ 用户程序中的块 ”一节）。
- 启动STEP 7
  从Windows用户界面启动STEP 7（见“ 启动STEP 7 ”一节）。
- 创建一个项目结构
  项目就像一个文件夹，所有数据都以分层的结构存于其中，任何时候你都可以使用。在创建了一个项目之后，所有其它任务都在这个项目下执行（见“ 项目结构 ”一节）。
- 组态一个站
  组态一个站就是指定你要使用的可编程控制器；例如，SIMATIC 300、SIMATIC 400、SIMATIC S5（见“ 插入站 ”一节）。
- 组态硬件
  组态硬件就是在组态表中指定你的自动化解决方案所要使用的模板以及在用户程序中以什么样的地址来访问这些模板。模板的特性也可以用参数进行赋值（见“ 组态硬件的基本步骤 ”一节）。
- 组态网络和通讯连接
  通讯的基础是预先组态网络。为此，要创建一个你的自动化网络所需要的子网，设置网络特性，设置网络的连接特性以及任何连网的站所需要的通讯连接（见“ 组态子网的步骤 ”一节）。

- 定义符号
  可以在符号表中定义局域或共享符号，在你的用户程序中用这些更具描述性的符号名替代绝对地址。（见“生成一个符号表”一节）。
- 创建程序
  用一种可供使用的编程语言创建一个与模板相连接或与模板无关的程序并以块、源文件或图表的形式存储（见“生成逻辑块的基本步骤”和“编写STL源文件的基本信息”两节）。
- 只适于S7：生成并评估参考数据
  利用这些参考数据可以使用户程序的调试和修改更容易（见“可用参考数据概述”一章）。
- 组态报文
  你可以生成与块相关的报文，比如包括它们文本和属性。使用传送程序，可以将生成的报文组态数据送至操作员接口系统数据库（例如SIMATIC WinCC。SIMATIC ProTool），（见“组态报文”一章）。
- 组态操作员控制和监视变量
  在STEP 7中可以生成操作员控制和监视变量并赋予它们所需的属性。使用传送程序，可以将生成的操作员控制和监视变量传至操作员接口系统WinCC的数据库（见“组态操作员控制和监视变量”一节）。
- 下载程序到可编程控制器
  只适于S7：完成所有的组态、参数赋值和编程任务之后，可以下载整个用户程序或其中的单个块到可编程控制器（你的硬件解决方案的可编程模板）。（见“下载条件”一节）。CPU已包含操作系统。
  只适于M7：从众多不同的操作系统中为你的自动化解决方案选择一个合适的操作系统，将该操作系统单独地或与用户程序一起传至M7可编程控制系统所需要的数据存储介质中。
- 测试程序
  只适于S7：进行测试，可以显示来自你的用户程序的变量数值，也可以是来自CPU的。对变量可以作赋值，为要显示或修改的变量生成一个变量表（见“用变量表进行测试的介绍”一节）。
  只适于M7：使用高级语言测试工具，测试用户程序。
- 监视操作，诊断硬件
  通过显示一个模板的在线信息，可以断定模板故障的原因。在诊断缓存区和堆栈内容的帮助下，可以判定用户程序处理中错误的原因。还可以检查一个用户程序是否可以在一个特定的CPU上运行（见“诊断硬件”和“显示模板信息”两节）。

- 制作设备文档
  在创建一个项目/设备后，为项目数据生成清楚的文档资料是非常有意义的，它使该项目的进一步编辑以及维护操作更容易（见“ 打印项目文档 ”一节）。DOCPRO是用来创建和管理设备文档的可选工具，使用该工具，你可以设计项目数据结构，译成接线手册的形式，并以通用格式打印出来。

特殊主题

当你创建一个自动化解决方案时，有些特殊主题可能会令你感兴趣。

- 多处理方式—多CPU的同步操作（见“ 多处理方式—多CPU的同步操作 ”一节）
- 多用户工作在一个项目中（见“ 多用户编辑项目”一节）
- 工作在M7系统（见“ M7系统程序的方法 ”一节）

1.2 STEP 7 标准软件包

使用标准

STEP 7 中集成的 SIMATIC 编程语言和语言表达方式，符合 EN 61131-3 或 IEC 1131-3 标准。标准软件包运行在操作系统 95/98/NT/2000 下，并与 Windows 的图形和面向对象的操作原理相匹配。

标准软件包的功能

标准软件支持自动任务创建过程的各个阶段，如：

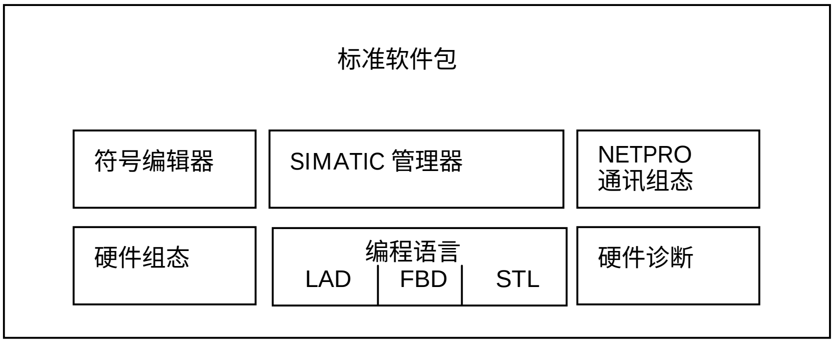
- 建立和管理项目
- 对硬件和通讯作组态和参数赋值
- 管理符号
- 创建程序，例如为S7可编程控制器创建程序
- 下载程序到可编程控制器
- 测试自动化系统
- 诊断设备故障

STEP 7 软件的用户接口，基于当前最新水平的人机控制工程设计，轻松使用。

STEP 7 软件产品手册在在线帮助和 PDF 格式电子手册中提供有所有在线信息。

STEP 7中的应用程序

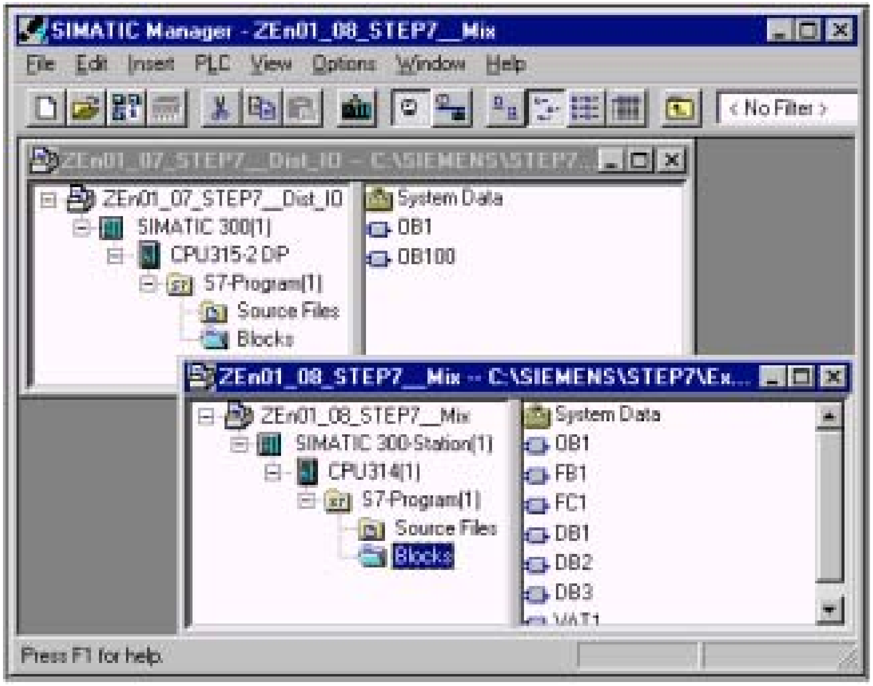
STEP 7 标准软件包提供一系列的应用程序（工具）：



无需分别打开各个工具；当选择相应功能或打开一个对象时，它们会自动启动。

SIMATIC 管理器

SIMATIC Manager（SIMATIC 管理器）可以管理一个自动化项目的所有数据，无论是为哪个可编程控制系统（S7/M7/C7）设计的。编辑所选数据所需要的工具由 SIMATIC Manager 自行启动。



符号编辑器

使用 Symbol Editor（符号编辑器），可以管理所有的共享符号。具有以下功能：

- 为过程信号（输入/输出）、位存储和块设定符号名和注释
- 分类功能
- 从/向其它的Windows程序导入/导出

使用这个工具生成的符号表可供其它所有工具使用。因而，对一个符号特性的任何变化都能自动被其它工具识别。

诊断硬件

这些功能可以向你提供可编程控制器的状态概况。这个概况中可以显示符号，指示每个模板是否正常或有故障。双击故障模板，可以显示有关故障的详细信息。信息的范围视各个模板而定：

- 显示关于模板的一般信息（例如，定货号、版本、名称）以及模板状态（例如，故障）
- 显示中央I/O和分布式从站的模板信息（例如，通道故障）
- 显示来自诊断缓存区的报文

对于 CPU，还可显示以下附加信息：

- 用户程序处理过程中的故障原因
- 显示循环时间（最长的、最短的和最近一次的）
- MPI的通讯可能性及负载
- 显示性能数据（可能的输入/输出、位存储、计数器、定时器和块的数量）

编程语言

用于S7-300和S7-400的编程语言梯形逻辑图（Ladder Logic）、语句表（Statement List）和功能块图（Function Block Diagram）都集成在一个标准软件包中。

- 梯形逻辑图（或LAD）是STEP 7编程语言的图形表达方式。它的指令语法与一个继电器的梯形逻辑图相似：当电信号通过各个触点、复合元件以及输出线圈时，使用梯形图，可以追踪电信号在电源示意线之间的流动。
- 语句表（或STL）是STEP 7编程语言的文本表达方式，与机器码相似。如果一个程序是用语句表编写的，CPU执行程序时则按每一条指令一步一步地执行。为使编程更容易，语句表已进行扩展，还包括一些高层语言结构（例如，结构数据的访问和块参数）。

- 功能块图（FBD）是STEP 7编程语言的图形表达方式，使用与布尔代数相类似的逻辑框来表达逻辑。复合功能（如数学功能）可用逻辑框相连直接表达。

其它编程语言作为可选软件包使用。

硬件组态

使用这个功能，可以为自动化项目的硬件进行组态和参数赋值。具有以下功能：

- 组态可编程控制器时，可以从电子目录中选择一个机架，并在机架中将选中的模板，安排在所需要的槽上。
- 组态分布式I/O与组态中央I/O一致，也支持以通道为单位的I/O。
- 在给CPU赋值参数的过程中，可以通过菜单的指导设置属性，比如，启动特性和循环扫描时间监控。支持多处理方式。输入的数据保存在系统数据块中。
- 在向模板作参数赋值过程中，所有可以设置的参数都是用对话框来设置的。没有任何设置使用DIP开关。参数赋值向模板的传送是在CPU启动过程中自动完成的。这意味着，例如，模板可以相互交换而无需赋值新的参数。
- 功能模板（FM）和通讯处理器（CP）的参数赋值，与其它模板的赋值方法一样，也是在硬件组态工具中完成的。对于每一个FM和CP，都有模板指定对话框和规则（包括在FM/CP功能软件包范围内）。通过只在对话框中提供有效的选项，系统可以防止不正确的输入。

NetPro（网络组态）

通过 MPI，可以实现使用 NetPro 时间驱动的循环数据传送，只要你在这里：

- 选择通讯的站
- 在表中输入数据源和数据目标；自动生成要下载的所有块（SDB），并且自动完整地下载到所有的CPU中。

事件驱动的数据传送也是可以实现的，只要你在这里：

- 设置通讯连接。
- 从集成的块库中选择通讯或功能块。
- 以你选择的编程语言为所选的通讯或功能块赋值参数。

1.3 STEP 7 版本 5.1 中的新内容

SIMATIC 管理器

- 使用ASCII 编辑器或表格编辑工具，并使用菜单命令Options > Managing Multilingual Text > Export，可以编辑非STEP 7项目的文本，以将项目翻译成其它国际语言。然后使用菜单命令Options > Managing Multilingual Text > Import，将文本导入STEP 7中。导出文件的格式为 “ *.csv ” 文件（使用逗号隔开）。
- 所有项目数据都可输入到CPU的专用存储卡中（新菜单命令PLC > Save Project to Memory Card和PLC > Get Project from Memory Card）。
- 使用菜单命令Options > Reference Data > Delete，可以删除当前参考数据。
- 使用菜单命令Help > About，用户可以读取其组件和DLL安装产品的版本信息。
- 在修改程序时，使用菜单命令Edit > Check Block Consistency，可以对块文件夹中的所有S7块进行一致性检查。由此，用户可以更好地测试接口修改对其它块的影响，快速排除错误。
- 用户可以采用系统属性导入新块，该属性即用户定义的用户程序块属性（例如，在上载系统库时）。每个块属性可以通过对话框调整。

LAD/STL/FBD编程块

- 使用新增菜单命令File > Check and Update Access，可以进行块的一致性检验。
- 当操作模式处于“ Test Operation（测试运行）”时，使用Monitor或使用Monitor with Call Path，用户可以监控所调用的块。为此，用户必须打开要调用的块，并将光标放在所要调用块上（STL中的CALL和LAD/FBD中的Callbox）。然后，点击鼠标右键，可以选择Called Block > Monitor命令和Called Block > Monitor with Call Path命令。
- 当删除块时，块符号也被删除。
  这就意味着如果相应的块已经从程序中删除，基本符号源不能再编译。
  当复制或移动块时，块符号仍保留。

监视和修改变量

- 监控和控制变量表已被修改。
  - 选择列。
  - 可以有多种选择。
  - 所有列都可以显示或隐藏。
  - 提供有工具提示，例如一个红色的行。

  可以编辑显示格式。
- 对话框“ Customize（自定义） ”有两个新的选项卡（“ General（普通） ”和“ Online（在线） ”）；在“ Online（在线） ”选项卡中，可以选择以下新选项：
  - 预选在线连接：至直接连接的CPU或组态的CPU。
  - 警告可以关闭。
  - “ Combine variables（组合变量） ”选项可以用于放大可监控的变量数量。
- 不用事先断开现有连接，就可改变连接。
- 在监控变量时，可以定义监控触发器。
- 通过选择行，并执行“ Modify（修改） ”功能，可以修改所选变量。只有高亮显示的变量才能修改。
- 提供有许多新增菜单命令，例如：
  - 打印预览（“ Table（表格） ”菜单）
  - 恢复重排（“ Window（窗口） ”菜单）
  - 建立至1、2、3、4的连接（“ PLC ”菜单；用于已用连接的快速修改）

组态和诊断硬件

- 当组态硬件时（新增菜单命令PLC > Monitor/Modify），可以监控/修改输入/输出。
- 新模块，例如，IM151/CPU可作为ET 200S系列智能DP从站。
- 智能DP从站的扩展组态能力：

  过程映象分区(部分过程映象)赋值,作为从站,使用直接数据交换,用于S7400 CPU；硬件中断OB赋值，用于PROFIBUS（用于通过DP主站用户程序控制中断可以触发硬件的I从站）
- 在线功能“ Module Information（模板信息） ”的人机工程学改进对于“ Diagnostic Buffer（诊断缓存区） ”选项卡，可以显示筛选结果（筛选出单个事件类）。

在“ Performance Data（性能数据） ”选项卡中，有关于组织块、系统块（SFC 和 SFB）以及地址区的信息概述。所有存储器的信息可以在“ Memory（存储器） ”选项卡中找到。

使用相应的监控时间来图形表示扫描循环时间，已改进为通过一个水平时间轴来表示。由此，可使辨别时间是超过还是没到作为参数赋值的监控时间，更容易。

组态网络和连接

- 连接表中的新增栏：局域接口和伙伴接口以及局域地址和伙伴地址。所有列都可单独显示或隐藏。因此，从连接表中可完全读取其连接路径，例如，可根据接口或子网分类。
- 当你退出项目时，将保存NetPro中的设置，当你再次打开项目时，又可获得其设置（甚至是其它编程器）。
- 由于子网是使用不同的颜色显示在显示屏上的，因此可以很容易地区分子网。

当你在打印输出一个图时，在打印属性对话框中可关闭子网颜色。

另外，使用缩放设置，可调整网络表示法，使之可适用更多页面。

- 除了PROFIBUS的总线参数外，也可打印输出其它子网（MPI）的参数。
- 支持新式WinAC插槽式CPU（CPU 41x2 DP PCI）的连接组态（S7连接）和连接状态。

参考数据

- 使用菜单命令Edit > Delete Symbols，你可以删除“ Use ”视窗中“ Symbols Not ”中没有使用的符号。
- 使用菜单命令Edit > Edit Symbols，你可以指定“ Addresses ”中所选地址的符号，无需“ Symbol ”视窗。
- 当你退出应用程序时，将保存窗口排列，不管显示的是什么视窗（交叉引用，程序结构等）。如果使用菜单命令Window > Save Arrangement When Exiting，可以恢复。

组态报文

- 你还可以生成用于M7程序的用户定义诊断报文。
- 用于编辑事件驱动通讯块的“PCS 7 Message Configuration（PCS 7报文组态）”对话框中包含两个寄存器，每个寄存器可最多输入10条报文。

CPU报文

- 在“CPU Messages（CPU报文）”应用程序中，有各种选项可编辑来报。
  使用菜单命令View > Automatic Shift，可在窗口中滚动浏览新来的报文，并选择报文。
  使用菜单命令View > Bring to the Foreground，可以将窗口至前台，显示报文。
- 使用菜单命令View > Leave in the Background，可显示报文，但窗口仍为后台方式。使用菜单命令View > Ignore Message， 在窗口中将不显示报文，也不归档。
- 使用菜单命令PLC > Remove Module，你可以从列表中清除高亮显示的模板。
- 使用“Settings CPU Messages（设置CPU报文）”对话框，你可以调整档案尺寸，保存登记模板列表，重新建立开始时的连接状态。另外，还可在ALARM_S/SQ显示报文文本。

报告系统错误

- 使用“Report System Error（报告系统错误）”功能，STEP 7 提供了一种极其方便的由报文形式的组件提供的诊断报文显示方法。STEP 7可以自动生成所需块和报文文本。你所需要做的就是将所生成的块装入CPU中，并将文本传送至所连接的HMI设备。你可查找各种Supported Components（支持组件）和Functional Scope（功能范围）下DP从站的诊断信息总览。

1.4 STEP 7 标准软件包的扩展应用

1.4.1 STEP 7 标准软件包的扩展应用

标准软件包可通过可选软件包进行扩展，这些可选软件包被分组为以下三个软件类别：

- 工程工具（Engineering Tool）；
  这些是较高层次的编程语言和面向工艺的软件。
- 运行版软件（Run-Time Software）； 这是用于生产过程的而无需架框的运行版软件。
- 人机接口（Human Machine Interface，HMI）；这是特别用于操作员控制和监视的软件。

下表是可选软件，你可以根据你的可编程控制系统来选用：

	STEP 7		
	S7-300 S7-400	M7-300 M7-400	C7-620
Engineering Tool			
· Borland C/C++		o	
· CFC	+[1)]	+	+[2)]
· DOCPRO	+	+[3)]	+
· HARDPRO	+		
· M7 ProC/C++		o	
· S7 GRAPH	+[1)]		+[2)]
· S7 HiGraph	+		+
· S7 PDIAG	+		
· S7 PLCSIM	+		+
· S7 SCL	+		+
· Teleservice	+	+	+
· Run-Time Software			
· Fuzzy Control	+		+
· M7-DDE Server		+	
· M7-SYS RT		o	
· Modular PID Control	+		+
· PC-DDE Server	+		

	STEP 7		
	S7-300 S7-400	M7-300 M7-400	C7-620
· PRODAVE MPI	+		
· Standard PID Control	+		+
Human Machine Interface			
· ProAgent			
· SIMATIC ProTool			
· SIMATIC ProTool/Lite			o
· SIMATIC WinCC			

O = 必须的

\+ = 可选的

1）=从 S7-400 以上建议使用

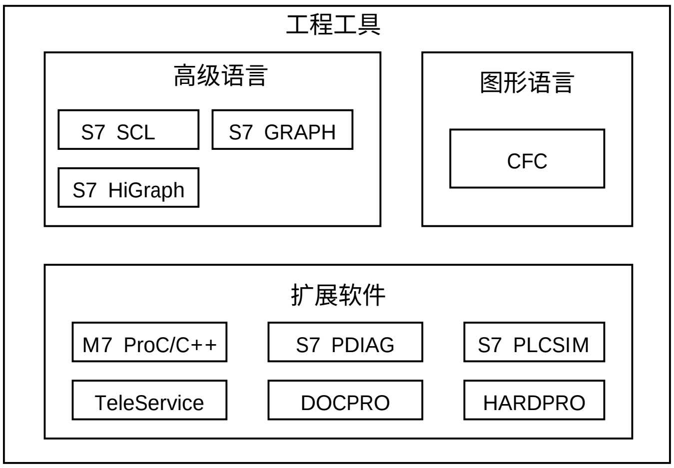
2） = 对于 C7-620 不推荐

3） = 不用于 C 程序

1.4.2 工程工具（Engineering Tool）

工程工具（Engineering Tool）是面向任务的工具，可被用于扩展标准软件包。工程工具（Engineering Tool）包括：

- 供编程人员使用的高级语言。
- 供技术人员使用的图形语言。
- 用于诊断、模拟、远程维护、设备文档制作等的扩展软件。



高级语言

下列语言作为可选软件包，用于 SIMATIC S7-300/S7-400 可编程控制器的编程：

- S7 GRAPH是用于编程顺序控制（步和转换）的编程语言。在这种语言中，过程顺序被分割为步。步中包含控制输出的动作。从一步到另一步的转换由转换条件控制。
- S7 HiGraph是以状态图的形式描述异步、非顺序过程的编程语言。为此，系统要被分解为几个功能单元，每个单元呈现不同的状态。各功能单元可通过在图形之间交换报文来同步。
- S7 SCL是符合EN 61131-3（IEC 1131-3）标准的高级文本语言。它包含的语言结构与编程语言Pascal和C相类似。所以S7 SCL特别适合于习惯于使用高级编程语言的人使用。例如，S7 SCL可以用于编程复杂或经常重复使用的功能。

图形语言

CFC 是用于 S7 和 M7 的编程语言，用来以图形方式连接已有的功能。这些功能涵盖了从简单的逻辑操作到复杂的闭环和开环控制等极为广泛的功能范围。大量的此种类型的功能在库中以块的形式提供。编程时将这些块拷贝到图表中并用线连接。

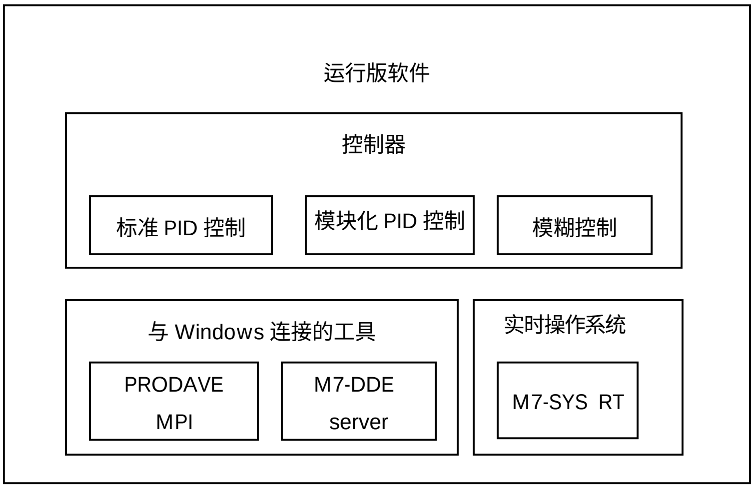
扩展软件

- Borland C++（只用于M7）包含Borland开发环境。
- 使用DOCPRO，可以将你用STEP 7生成的全部组态数据构造为接线手册。这使得组态数据的管理更为容易，并且可以为按照指定标准的打印准备好信息。
- HARDPRO是S7-300硬件组态系统，它支持用户对复杂的自动化任务的大范围的组态。
- M7-ProC/C++（只用于M7），允许将编程语言C和C++的Borland开发环境集成到STEP 7的开发环境中。
- 你可以使用S7 PLCSIM（只用于S7）模拟将S7可编程控制器连接到编程设备或PC，以便进行测试。
- 使用S7 PDIAG（只用于S7），可以标准化组态SIMATIC S7-300/S7-400过程诊断。使用过程诊断，可以检测可编程控制器之外的故障和故障状态（例如，未到达限位开关）。
- 使用TeleService，就可以使用编程设备或PC，通过电话网对S7和M7可编程控制器作远程编程和服务。

1.4.3 运行版软件

运行版软件包括可以由用户程序调用的预编程的解决方案。运行版软件直接集成在自动化解决方案中。它包括：

- SIMATIC S7控制器，例如，标准模板，以及模糊控制。
- 用于连接可编程控制器和Windows应用程序的工具。
- SIMATIC M7的一个实时操作系统。



SIMATIC S7 控制器

- 使用标准PID控制，可以将连续控制器、脉冲控制器以及步进控制器集成到用户程序中。使用带有集成控制器设置的参数赋值工具，可以让你在一个很短的时间内将控制器设为最优使用。
- 如果简单的PID控制器不足以解决你的自动化任务，则可以使用模块化PID控制。通过在提供的标准功能块中作连接，几乎可以设计和建立任何控制结构。
- 使用模糊控制可以生成模糊逻辑系统。当过程很难或无法用数学模型来描述；过程特性不可预知；或者出现非线性；但具有过程运行的经验时，可以使用这种模糊系统。

用于连接Windows的工具

- PRODAVE MPI是用于SIMATIC S7、SIMATIC M7和SIMATIC C7之间过程数据通信的工具箱。它通过多点接口（MPI）自行管理数据通信。
- 使用M7-DDE服务器（动态数据交换），无需另外编程就可将Windows应用程序连接到SIMATIC M7的过程变量。

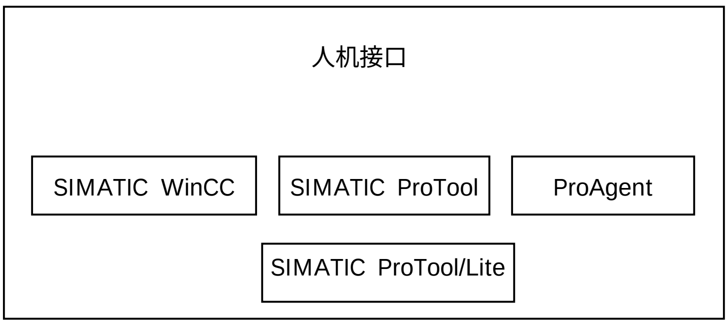
实时操作系统

- M7-SYS RT包含M7 RMOS 32操作系统和系统程序。它是SIMATIC M7软件包使用 M7-ProC/C++和CFC的前提条件。

1.4.4 人机接口

人机接口（HMI）是专门用于 SIMATIC 中操作员控制和监视的软件。

- 开放的过程监视系统SIMATIC WinCC，是一个基本的操作员接口系统，它包括所有重要的操作员控制和监视功能，这些功能可以用于任何工业系统和使用任何工艺。
- SIMATIC Pro Tool和SIMATIC ProTool/Lite是用于组态SIMATIC操作员面板（OP）和SIMATIC C7紧凑型设备的现代工具。
- ProAgent通过建立有关故障原因和位置的信息可实现对系统和设备的有目的快速过程诊断。



2 安装与授权

2.1 授权

2.1.1 授权

你需要产品特别授权（用户权）以使用 STEP 7 编程软件。因此，本软件为拷贝保护设计，并且仅当在相应的编程器或 PC 的硬盘上程序或软件包的相关授权被找到才能使用。

例如，STEP 7 与可选软件包，需要不同的授权。

2.1.2 安装与取出授权

授权磁盘

在软件供货范围中，包括一张只读的授权磁盘。它含有实际授权。含有 STEP 7 V5.1 的 CD-ROM 上的程序“ AuthorsW ”，用于显示、安装和取出授权。

授权的数量由授权磁盘上的授权记数器决定。你每次安装一个授权，该记数器减“ 1 ”。当记数器值到达“ 0 ”时，你不可能用这张磁盘再安装任何授权。

提示

你收到的黄色授权磁盘带有 STEP 7 标准软件的相应授权。

对于可选软件包，你将收到一张带有为此的一个授权的红色授权磁盘。

注意

阅读授权磁盘上 README.TXT 文件中的提示和“ Guidelines for Handling Authorizations ”中的准则。如果你不坚持这些准则，可能会不可补地丢失授权。

没有授权，你也可以使用标准软件以便熟悉用户接口和功能。然而只有安装了授权才允许有效地使用 STEP 7 工作。如果你还未安装授权，你将每隔一段时间被敦促安装它。

如果你丢失授权

授权可能丢失，例如，如果硬盘出现故障并且你没有机会从故障的硬盘取出授权。

如果你丢失授权，你可以使用紧急授权。它也包含在授权磁盘上。该紧急授权允许你继续运行该软件一段有限时间。在这种情况下，在有效的时间超出之前，当你起动时屏幕上显示出所剩的时间。在此期间你应获得丢失授权的替换授权。为此，与你当地西门子代表处联系。

提示

当你安装紧急授权的同时，有效期即开始生效，即使你不起动 STEP 7。即使你把授权写回磁盘，紧急授权也不停止运行。

安装AuthorsW

用于显示、安装和取出授权的“ AuthorsW ”程序，也在含有 STEP 7 V5.1 的 CD-ROM 上。你使用安装（Setup）程序把该程序安装在你的硬盘上，从这里你就可以用它进行授权操作。

提示

AuthorsW 程序的默认位置是“ START > SIMATIC > AuthorsW > AuthorsW ”。

在你第一次安装时安装授权

当第一次安装 STEP 7 软件时，当信息提示你安装授权时，你应按如下进行：

1. 当提示时，把授权磁盘插入驱动器。
2. 应答提示。
3. 授权被传送到一个物理驱动器。

在以后的日子里增加一个授权

如果你试图起动没有效授权的 STEP 7 软件时，将出现信息告诉你这一点。为了接下来增加授权，步骤如下：

1. 把授权磁盘插入软盘驱动器，例如，驱动器 A。
2. 从硬盘起动程序“ AUTHORSW.EXE ”。

3. 在两个目录框中的一个目录框中选择含有授权的驱动器，在另一个中选择目标驱动器。显示两个驱动器上的所有现有授权。
4. 选择所需授权。
5. 点击按钮“ <-- ”或“ --> ”。所选择的授权将被传送到所选驱动器中。
6. 关闭该对话框。

提示

如果 Windows NT 能完全访问硬盘驱动器“ C ”和目标驱动器，授权才有效。

授权升级

使用菜单命令“ Upgrade ”来升级授权。为此，你需要：

- 你要升级的授权的授权磁盘，
- 硬盘上的授权程序“ AuthorsW，
- 磁盘上的新的STEP 7 Upgrade，
- 磁盘或硬盘上的旧授权。

在升级的过程中，旧授权被删除，并换成新授权。因此，授权磁盘在任何时候都不应是写保护的（只读）。

1. 插入新授权磁盘。
2. 从硬盘起动程序“ AUTHORSW.EXE ”。
3. 选择菜单命令 AuthorigatIon＞Upgrade。对话窗出现，在此你可以选择升级程序。然后提示你插入带有旧授权的授权磁盘。
4. 插入所需的授权磁盘。然后提示你是否真的想升级。这是中断这个过程的最后机会。一旦你确认了对话窗，这个过程必须不被任何因素所中断，否则授权将丢失。
5. 使用“ OK ”按钮确认。接着将提示你插入带有新授权的授权磁盘。

然后检查所有必要的条件。当成功地完成检查后，新的授权将被激活，并完成升级。

新的授权将安装在旧授权所安装的驱动器上。如果必要的话，从软盘安装授权至硬盘。

恢复授权

如果你的授权出毛病，与热线联系。然后你就能够用菜单命令 Authorization > Restore 恢复授权。

取出授权

如果你需要重复授权，例如，如果你要重新格式化授权所在的驱动器，你就必须首先取回授权至一张西门子授权盘（卸载）。使用该授权磁盘，你也可以把所用的可选软件包的授权取回。

把授权传回授权磁盘，步骤如下：

1. 把授权磁盘插入软盘驱动器，例如，驱动器 A。
2. 从硬盘起动程序“ AUTHORSW.EXE ”。
3. 在两个目录框中的一个目录框中选择授权所安装的驱动器，在另一目录框中选择目标驱动器。显示两个驱动器上的所有授权。
4. 选择你所需的授权。
5. 点击“ ← ”或“ → ”按钮。所选授权就被传送到授权磁盘或所选驱动。
6. 如果你不想再移动任何授权，关闭对话窗口。以后你可以用这张磁盘再次安装一个授权。

你也可以在硬盘（网络驱动器）之间移动授权。

2.1.3 管理授权的原则

注意

阅读本章中和授权盘上的“ README.TXT ”文件中的提示。如果你不坚持这些准则，可能会不可补地丢失授权。

什么时候需要取出授权?

当你格式化、压缩或恢复你的硬盘之前，或当你安装新的操作系统之前，你必须取出任何已有的授权。

备份

如果你备份含有授权拷贝的硬盘，当你把你备份的数据恢复到你的硬盘时，这些拷贝可能会覆盖掉有效的授权，结果破坏了授权。

为避免有效的授权被备份的拷贝覆盖，你必须如下去做：

- 在你做备份拷贝之前取出所有授权。
- 从备份中去掉授权。

优化你的硬盘

如果你使用能够提供移动固定数据块的优化程序，仅当你把所有的授权都从硬盘传送到授权磁盘后才能使用。

故障扇区

当你安装授权时，目标盘上某磁道有时被标为“ 故障 ”。不要试图修理该磁道。否则你可能会破坏授权。

写保护和拷贝保护

授权磁盘必须不是写保护的。

可以拷贝授权磁盘上的文件到另一驱动器（例如，硬盘），并在那里使用。然而，不能使用这些拷贝的文件授权，你需要原始的授权磁盘来授权。

允许的驱动器

只能把授权安装在硬盘上。对压缩的驱动器（例如，DBLSPACE），你可以把授权安装在相应的主驱动器上。

授权工具可以防止授权被安装在非法的驱动器上。

存贮位置

当你安装授权时，授权文件存贮在带有“ （System）系统 ”和“ （Hidden）隐藏 ”属性的保护子目录“ AX NF ZZ ”中。

- 这些属性禁止改变。
- 禁止修改或删除文件。

- 禁止移动该子目录。从该子目录（授权）拷贝的文件，会被识别为故障，并且使授权失效。

否则，授权将不可恢复地丢失。

保护的子目录“ AX NF ZZ ”在每个驱动器被生成一次，并包含安装在该驱动器上的所有授权，当第一个授权被安装时，它就被生成，并当最后一个授权被取出时，被删除。

对每个授权，将在该保护的子目录中，生成两个名字相同但扩展名不同的文件。这些文件被给予同授权相同的名字。

授权的数目

只要存贮空间足够，用户可以在驱动器上安装任意多个自己所需要的授权，但是相同的版本只能有一个授权（例如：STEP 7 V 4.x 只能有一个、 STEP 7 V5.x 只能有一个）。这些授权彼此不会发生冲突。

破坏的授权

在硬盘驱动器上损坏的授权，不能被 Authorsw 程序删除。这些损坏的授权甚至有可能妨碍用户安装新的及有效的授权。在这种情况下，请与当地西门子代表处联系。

授权工具

当前所用的授权工具 AuthorsW 版，比任何旧版本功能更强。

提示

如果使用 V2.0 版，不能识别所有旧的授权，在这种情况下，用户应使用更老的 AUTHORS 版（DOS 版）<V3.x 以下的授权工具来工作。

2.2 安装 STEP 7

2.2.1 安装 STEP 7

在 STEP 7 中，安装有 Setup 程序，使用该程序，可自动地进行安装。用户可按照屏幕弹出的指南信息的引导，一步一步地完成整个安装步骤。用户可用标准的 Windosw 95/98NT 或 Windows 2000 软件安装功能，调用 Setup 程序。

主要安装步骤：

- 将数据拷贝到你的编程器上
- 设置EPROM和通讯的驱动器
- 输入ID号
- 授权（如果需要）

提示

西门子编程器（如 PG 740）已将 STEP 7 软件装在硬盘上，只需释放安装即可。

安装要求

- 操作系统：

  Microsoft Windosw 95，Windows98，Windows 2000 或 Windows NT。
- 基本硬件：

  编程器或 PC：

  - 80486 处理器以上（Windosw NT 要求奔腾处理器）和
  - RAM：至少 32 Mbyles，建议 64 Mbytes
  - Microsoft Windows 支持的彩色显示器、键盘和鼠标

编程器（PG）是专门为在工业环境中使用而设计的个人计算机。它安装了用于 SIMATIC 可编程序逻辑控制器编程时所需的一切。

- 存贮能力：

  请参考 readme 文件中提供的所需硬盘空间的要求。

- 多点接口（可选）：

  如果用户需要在 STEP 7 中用多点接口（MPI）方式与可编程序逻辑控制器进行通讯，则在编程器或 PC 与可编程序控制器之间只需要多点接口（MPI）。

  因此，用户需要：

  - 在计算机的通讯口上连接 PC/MPI 电缆，或
  - 在计算机中安装 MPI 卡

编程器中已经装有多点接口。

- 外部接口（可选）

  如果用户希望在 PC 上为 EPROM 编程，则还需要一个外部接口。

提示

见 README.WRI 文件中的 “ 安装 STEP 7 ” 的注意事项和 “ 与 STEP 7 基本软件包兼容的 SIMATIC 软件包清单 ” 。

使用命令 Start > Simatic > Product Notes，在 “ Start（开始） ” 菜单中，可以找到 Readme 文件。

使用命令 Start > Simatic > Documentation，在 “ Start（开始） ” 菜单中，可以找到兼容性列表。

2.2.2 安装步骤

安装准备

在用户开始安装软件之前，必须先启动 Windows 95/98/NT/2000。

- 如果在用户编程器的硬盘上已装有STEP 7软件，则用户不再需要任何外部数据媒介。
- 从磁盘上安装STEP 7时，应先将第1张盘插入到用户编程器或PC的软驱中。
- 从光盘上安装时，要将光盘放入用户PC机的光驱中。

开始安装程序

按如下步骤安装软件：

1. 插入磁盘（磁盘 1）或光盘，双击文件“ SETUP.EXE ”，起动安装程序。
2. 一步一步地按照安装程序所显示的指令进行。

在整个安装过程中，安装程序一步一步地指导用户如何进行。在安装的任何阶段，用户都可以切换到下一步或上一步。

在安装过程中，在对话框中显示一些询问需要用户回答，还有一些选项需要用户选择。请阅读下列提示，可帮助你既快又容易地回答一些安装访问。

如果已经安装了STEP 7的某一种版本...

如果安装程序发现，在编程器上已有另一版本的 STEP 7，它将报告该情况，并提示用户如何进行：

- 中断安装，以便用户可以将旧的STEP 7版本在Windows下卸载，然后，再开始安装。
- 或者，继续安装，用新版本覆盖旧版本。

如果用户在安装新版本之前，卸载旧版本，则用户软件能够较好地组织。使用新版本覆盖旧版本有一个缺点，就是，如果以后做卸载时，旧版本中保留的部分，不能被删除。

选择安装选项

在用户选择安装范围时，有三种选项：

- 标准组态：用于用户接口的所有语言、所有应用以及所有的举例。请参考最新产品信息中对这种组态所要求的存储空间。
- 最小组态：只有一种语言，没有举例。请参考最新产品信息中对这种组态所要求的存储空间。
- 用户定义组态：用户可定义安装范围，选择用户希望安装的程序、数据库、举例和通讯功能。

ID号码

在安装的过程中，将提醒用户输入一个 ID 号码。要求输入的 ID 号码，可在软件产品证书中或授权盘中找到。

使用授权

在安装过程中，安装程序将检查硬盘上是否有授权。如果没有发现授权，会出现一条信息，指出该软件只能在有授权的情况下使用。如果用户愿意，可立即运行授权程序，或者继续安装，稍后再执行授权程序。在前一种情况中，应插入授权盘。

PG/PC接口设置

在安装过程中，会出现一个对话框，在这个对话框中，用户可以设置 PG/PC 接口的参数。用户可在 “ 设置 PG/PC 接口（Setting the PG/PC Interface） ” 中找到更多信息。

设置存储卡参数

在安装过程中，会出现一个对话框，在这个对话框中，用户可以为存储卡分配参数。

- 如果用户不用存储卡，则不需要EPROM驱动器.选择 “ NO ERROM Driver ” 选项。
- 否则，选择应用到你的编程器上的输入路径。
- 如果用户使用的是PC，则可选择用于外部编程口的驱动器。这里，用户必须定义哪个接口用于连接编程口（例如，LPT1）。

在安装完成之后，用户可通过 STEP 7 程序组或控制面板中的 “ Memory Card Parameter Assignment（存储卡参数赋值） ”，修改这些设置参数。

闪存文件系统

在为存储卡赋参数的对话框中，用户可以定义是否应安装闪存文件系统。

例如，当用户需要在 SIMATIC M7 中向 EPROM 存贮卡中写入某些文件，或从 EPROM 存贮卡中删除某些文件，同时，不改变存储卡中保留的内容时，就需要有闪存文件系统。

如果用户使用某个适合的编程器（PG720/PG740/PG760）或用外部编程口，并且，希望使用此功能，则选择闪存文件系统的安装。

如果在安装过程中出现错误

下列错误可能导致安装失败：

- 如果在起动后，立即出现一个初始化错误，该程序很可能不能在Windows下起动。
- 没有足够的存储空间：对于标准软件，不考虑用户安装的范围，在硬盘上至少需要100Mbytes的空间。
- 坏盘或坏光盘：如果盘是坏的，请与当地西门子代表处联系。
- 操作员错误：重新安装，并仔细阅读各项指令。

完成安装

如果安装成功，会在屏幕上出条信息告知用户。

如果在安装的过程中，改变了系统文件，将建议你重新起动 Windows。当用户完成这些以后，可以开始基本的 STEP 7 应用，SIMATIC 管理器。

一旦安装成功完成，会为 STEP 7 生成一个程序组。

2.2.3 设置 PG/PC 接口

通过这里所做的设置，用户可以设置编程器/PC 与可编程序控制器之间的通讯连接。在安装过程中，会出现一个对话框，在这个对话框中，用户可以设置 PG/PC 接口的参数。在安装之后，用户可在 STEP 7 程序组中调用“ Setting PG/PC 接口 ”程序，显示该对话框。这将使你在安装之后，可以改变接口参数。

基本程序

为了对接口进行操作，用户需要如下步骤：

- 在操作系统中设置
- 适合的接口参数

如果用户使用编程器并通过多点接口（MPI）进行连接，则不再需要其它的操作系统特别适配方法。

如果用户使用 PC 机和 MPI 卡或通讯处理器（CP），则应检查在 Windows 中“ Control Panel（控制面板）”里的中断和地址设置，以确保没有中断冲突和地址区重叠。

为了使向编程器/PC 接口分配参数容易进行，在显示的对话框中提供一套预先定义的基本参数（接口参数）供用户选择。

为PG/PG接口分配参数

为了设置模板参数，请按照下列步骤要点进行（可在在线帮助中找到详细描述）：

1. 双击“ Control Panel（控制面板）”中的“ Setting PG/PC Interface（设置 PG/PC 接口）”。
2. 将“ Access Point of Application（应用访问点）”设置为“ S7ONLINE。”
3. 在“ Interface Parameter set used（所用接口参数集）”的表中，选择所需接口参数赋值。如果没有显示你所需要的接口参数，用户必须用“ Select（选择）”按钮先安装模板或协议。然后，接口参数会自动生成。
   - 如果用户所选的接口能自动识别总线参数（例如：MPI-ISA Card（Auto）），则用户可以直接将编程器或 PC 至 MPI 或 PROFIBUS 上，而不需要设置总线参数。如果传输率小于 187.5Kbps，在读总线参数时，有可能有将近 1 分钟的延迟。自动识别条件：主站分配循环总线参数并连接到总线上。所有与此有关的新 MPI 组件，必须使能总线参数的循环分配（缺省 PROFIBUS 网络设置）。
   - 如果用户选择接口为不自动识别总线参数，则用户可以显示该属性，并将这些总线参数与子网适配。

如果与其它设置发生冲突，需要做必要的修改（例如，修改中断或地址的设置）。在这种情况下，在 Windows 的硬件组态和控制面板中做相应的修改（见下面）。

注意

禁止删除模板参数设置“ TCP/IP ”（如果“ TCP/IP ”参数出现的话）。它将防止其它应用功能不会被修改。

检查中断和地址设置

如果用户使用的是PC带有MPI卡，则应检查缺省设置的中断和地址区是否被占用，如果需要，选择一个没被占用的中断和地址区。

用户可在 Windows95/98 下显示当前的设置，步骤如下：

1. 在“Contral Panel（控制面板）”中打开“System（系统）”对话框，并选择“Device Manager（设备管理器）”选项卡。
2. 在所显示的表中选中“Computer（计算机）”，并点击“Properties（属性）”按钮。
3. 在另一个对话框中，用户通过选择相应的选择按钮，可以显示所占用的中断列表（IRQ）或所占用的地址区（I/O）。

在 Windows NT 下，用户可以：

- 在Start > Programs > Administrative Tools（Common） > Windows NT/2000 Diagnostics > Resources下，显示资源设置。
- 在PG/PC Interface > Install > Resources下，修改资源。

在 Windows 2000 下，用户可以：

- 在Control Panel > Administrative Tools > Computer Management > System Tools > System Information > Hardware Resources下，显示资源设置。

Windows 9x 与Windows NT/2000之间的不同之处

用户必须在 Windows NT/2000 中的一个特殊的对话框中设置中断、地址区和其它资源（详细描述请参阅在线帮助）。

2.3 卸载 STEP 7

2.3.1 卸载 STEP 7

使用通常的 Windows 步骤来卸载 STEP 7：

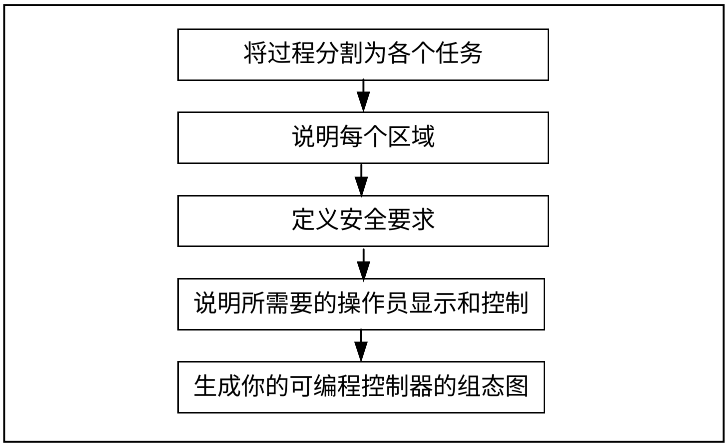
1. 在“ Control Panel ”中，双击“ Add/Remove Programs ”图标，启动 Windows 下用于安装软件的对话框。
2. 在安装软件显示的项目表中，选择 STEP 7。点击“ Add/Remove（加入/删除软件）”按键。
3. 如果“ Remove Enabled File（删除使能的文件）”对话框出现，如果用户不知如何回答，则可点击“ No ”按键。

3 设计自动化解决方案

3.1 设计一个自动化项目的基本步骤

本章概述了为一个可编程控制器（PLC）设计一个自动化项目所涉及的基本任务。基于一个自动工业搅拌过程示例的指导，将一步一步贯穿整个过程。

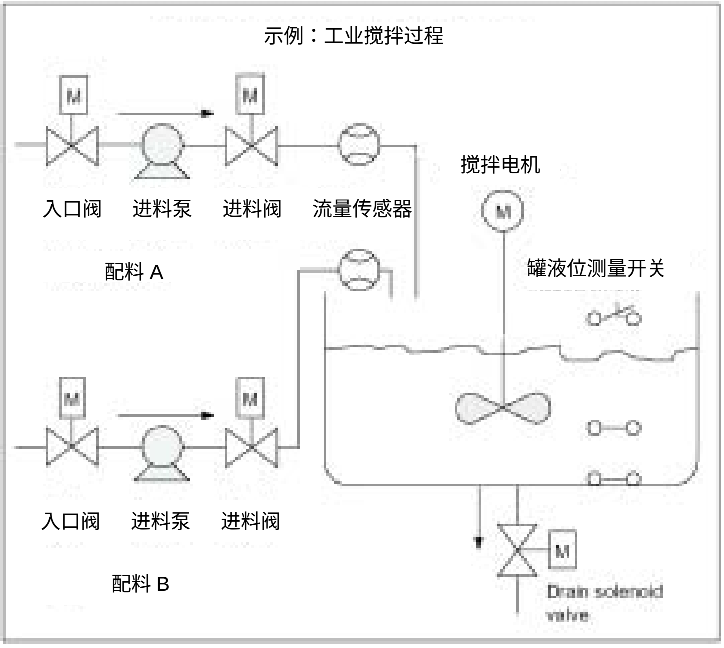
设计一个自动化项目的方法有很多。可用于任何项目的基本步骤的说明如下图所示。



3.2 将过程分割为任务和区域

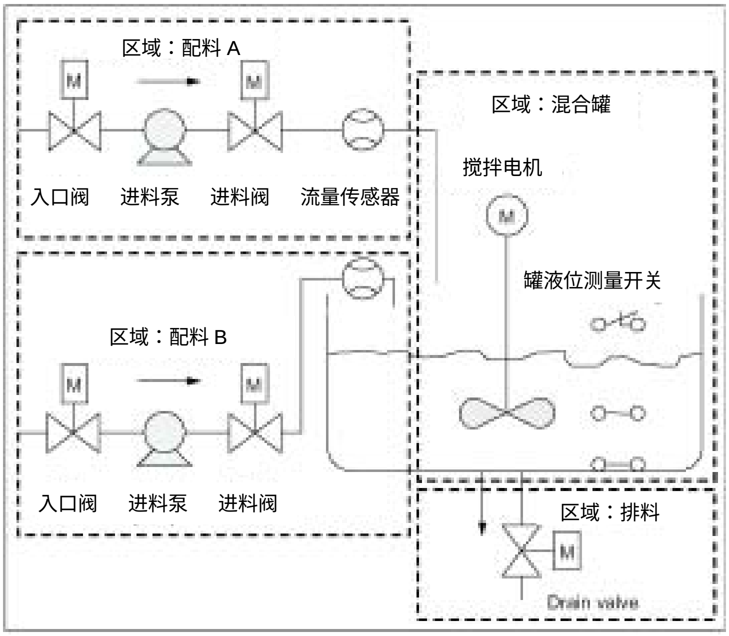
一个自动化过程包括许多单个的任务。通过识别一个过程内的相关任务组，然后将这些组再分解为更小的任务，即使最复杂的过程也能够被定义。

下面这个工业搅拌过程的例子可以用来说明如何将一个过程构造为功能区域和单个的任务：



决定过程的区域

在定义了要控制的过程之后，将项目分割成相关的组或区域：



由于每组被分为小任务，所以控制过程在这一部分所要求的任务就不那么复杂了。

在我们的工业搅拌过程示例中，你可以看到四个不同的区域（见下表）。在这个例子中，配料 A 的区域中包含的设备与配料 B 的区域相同。

功能区域	使用的设备
配料 A	配料 A 的进料泵 配料 A 的入口阀 配料 A 的进料阀 配料 A 的流量传感器
配料 B	配料 B 的进料泵 配料 B 的入口阀 配料 B 的进料阀 配料 B 的流量传感器
混合罐	搅拌电机 罐液位测量开关
排料	排料阀

3.3 说明各个功能区域

当你说明过程中的各个区域和任务时，不仅要定义每个区域的操作，而且要定义控制该区域的各种组件。这包括：

- 每个任务的电的、机械的和逻辑的输入和输出
- 各个任务的互锁和相关性

本示例工业搅拌过程使用泵、电机和阀门。必须对这些设备作精确描述，以识别其操作特性和操作过程所要求的互锁类型。下表提供的示例是对工业搅拌过程中使用的设备的描述。完成说明后，还可以用它来订购所需要的设备。

配料 A/B：进料泵电机
进料泵电机传送配料 A 和 B 到混合罐 - 流速：400 升（100 加仑） /分钟 - 速率：1200 转/分时，100 千瓦（134 马力）
泵由混合罐附近的操作员站控制（启动/停止）。启动的次数被计数以便进行维护。计数器和显示都可以由一个按钮复位。
对泵进行操作必须满足以下条件： - 混合罐不满 - 混合罐的排料阀关闭 - 紧急关断未动作。
如果满足下列条件，则泵被关断： - 在泵电机启动 7 秒后流量传感器仍指示没有流量 - 流量传感器指示流动已停止

配料 A/B：入口阀和进料阀
配料 A 和 B 的入口阀和进料阀可以允许或防止配料进入混合槽。
阀门是带有弹簧的螺线管 - 如果螺线管动作则送出阀打开。 - 如果螺线管不动作则送出阀关闭。
入口阀和进料阀都由用户程序控制。
满足以下条件，排料阀可以打开： - 进料泵电机至少运行 1 秒。
如果满足下列条件，则泵被关断： - 流量传感器指示没有流量

搅拌电机
搅拌电机在混合罐中混合配料 A 和配料 B - 速率：1200 转/分时 100 千瓦（134 马力）
搅拌电机由混合罐附近的操作员站控制（启动/停止）。启动的次数被计数以便进行维护。计数器和显示都可以由一个按钮复位。
对泵进行操作必须满足以下条件： - 罐液位传感器没有指示“ 罐液位低于最低限 ” - 混合罐的排料阀是关闭的 - 紧急关断未动作：
如果满足下列条件，则泵被关断： - 在电机启动后的 10 秒内转速计未指示已达到额定速度

排料阀
排料阀让混合物排出（靠重力排出）到过程的下一阶段。阀门是带有弹簧的螺线管 - 如果螺线管动作则送出阀打开 - 如果螺线管不动作则送出阀关闭
送出阀由一个操作员站控制（打开/关闭）。
以下条件满足排料阀可以打开： - 搅拌电机关断 - 罐液位传感器未指示“ 罐空 ” - 紧急关断未动作：
如果满足下列条件，则泵被关断： - 罐液位传感器指示“ 罐空 ”。

罐液位测量开关
混合罐中的开关指示罐的液位高度关用来联锁进料泵和搅拌电机

3.4 列表输入，输出和入/出

为每个要控制的设备写出物理说明后，为每个设备或任务区域画出输入和输出图。



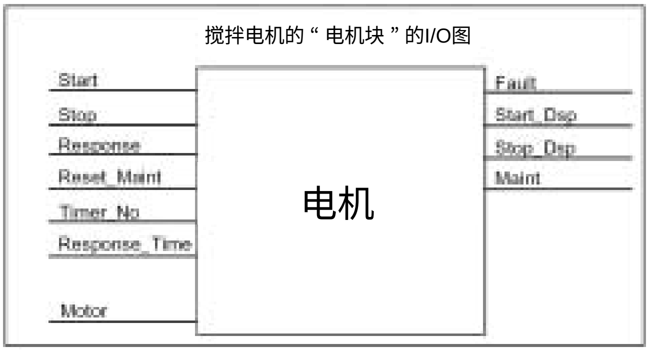
这个图相应于要编程的逻辑块。

3.5 为电机生成一个 I/O 图

在我们这个工业搅拌过程的例子中使用了两个进料泵和一个搅拌电机。每个电机由它自己的 " 电机块 " 控制，而这个 " 电机块 " 对三个设备都是一样的。该块需要六个输入：两个用于启动或停止电机，一个用于复位维护显示，一个用于电机的响应信号（电机运行/未运行），一个用于运行期间必须接收的响应信号，一个用于流量时间的定时器的号码。

逻辑块还需要 4 个输出：两个指示电机的操作状态，一个指示故障，一个指示电机应维护了。

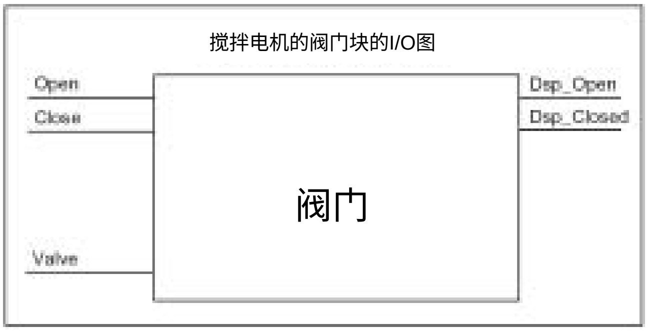
还需要一个入/出参数启动电机。它被用作控制电机但同时也在 " 电机块 " 的程序中被编辑并修改。



3.6 为阀门创建一个 I/O 图

每个阀由它自己的“阀门块”控制，该块对所有的阀都是一样的。该逻辑块有两个输入：一个用来打开阀，一个用来关闭阀。它还有两个输出：一个用于指示阀是打开的，另一个用于指示阀是关闭的。

该块有一个入/出参数用于启动该阀。它被用作控制阀门但同时也在“阀门块”的程序中被编辑和修改。



3.7 建立安全要求

根据法定的要求及公共健康和安全政策，决定为确保过程安全还需要哪些附加组件。在你的描述中还应包括那些安全组件对你的过程区域的任何影响。

定义安全要求

确定哪些设备需要硬件接线电路以达到安全要求。通过定义，这些安全电路的操作独立于可编程控制器之外（虽然安全电路通常提供一个 I/O 接口以便与用户程序相配合）。通常你要组态一个矩阵来连接每一个执行器，这些执行器都有它自己的紧急断开范围。这个矩阵是安全电路的电路图的基础。

要设计安全机制可按如下进行：

- 决定每个自动化任务之间逻辑的和机械的/电的互锁。
- 设计电路使得属于过程的设备可以在紧急情况下手动操作。
- 为过程的安全操作建立更进一步的安全要求。

建立安全电路

在工业搅拌过程示例中使用了以下逻辑作为它的安全电路：

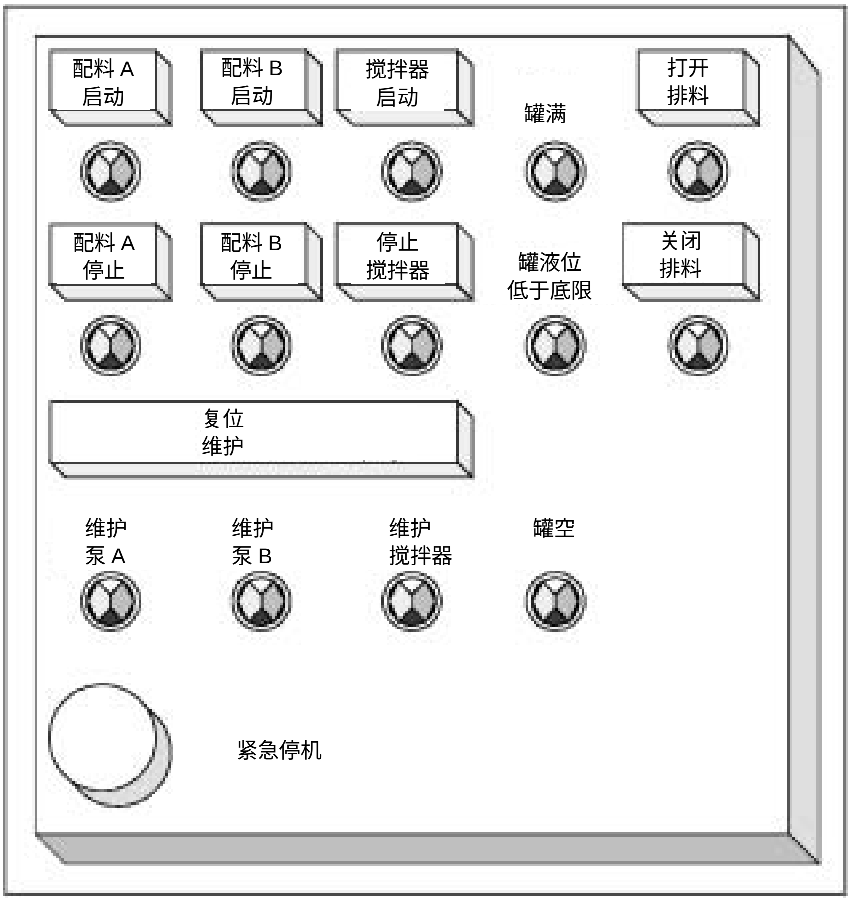
- 一个紧急断开开关可独立于可编程控制器（PLC）之外关掉以下设备：
  - 配料A的进料泵
  - 配料B的进料泵
  - 搅拌电机
  - 阀门
- 位于操作员站的紧急断开开关。
- 一个用于指示紧急断开开关状态的控制器的输入。

3.8 描述所需要的操作员显示和控制

每个过程需要一个操作接口，使得操作人员能够对过程进行干预。设计技术规范的部分包括操作员控制站的设计。

定义操作员控制站

在我们示例中所描述的工业搅拌过程，每个设备都可以由操作员控制站上的按钮来启动或停止。这个操作员控制站包括用以指示操作状态的指示灯（见下图）。



操作站上还包括指示设备在经过一定次数的启动后需要维护的指示灯以及可以使过程立即停止的紧急断开开关。操作站上还有用来复位三个电机的维护显示灯的按钮。用这个按钮可以关断用于指示电机应进行维护的维护指示灯并将相应的计数器清零。

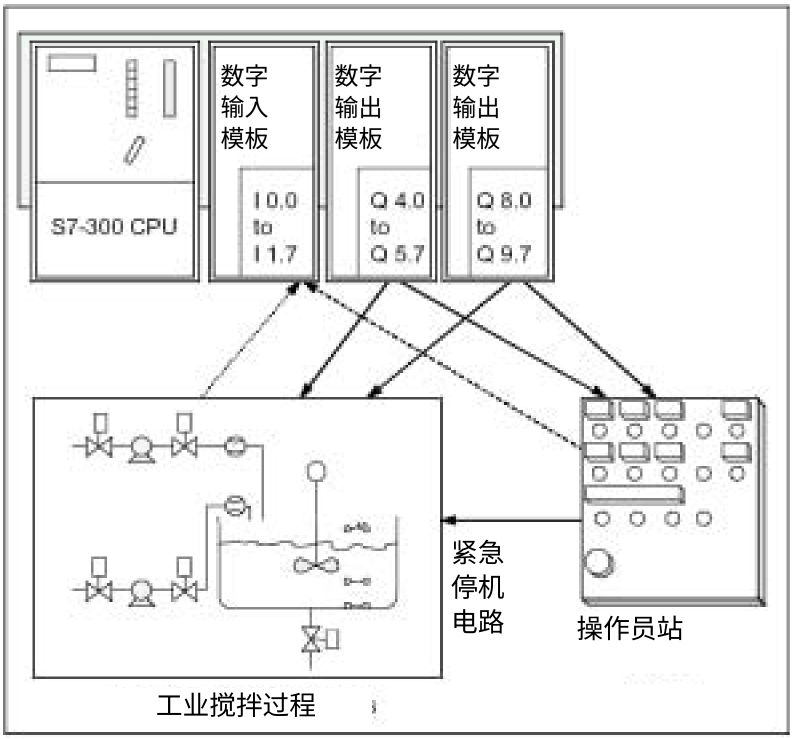
3.9 生在一个组态图

在制作了设计要求的文档后，还必须决定项目所需的控制设备的类型。

通过决定使用什么样的模板也就指定了可编程控制器的结构。生成一个组态图指定以下方面：

- CPU类型
- I/O模板的类型及数量
- 物理输入和输出的组态

下图所示为工业搅拌过程的 S7 组态的示例。



4　设计程序结构基础

4.1　CPU 中的程序

在一个 CPU 中，有两种不同的程序总会被执行：

- 操作系统
- 用户程序

操作系统

每个 CPU 都有一个操作系统，用以组织与特定的控制任务无关的 CPU 的功能和顺序。操作系统的任务包括以下各项：

- 处理暖起动和热起动
- 刷新输入的过程映象表并送出输出的过程映象表
- 调用用户程序
- 检测中断并调用中断OB
- 检测并处理错误
- 管理存储区域
- 与编程设备和其它通讯伙伴之间的通讯。

如果修改了操作系统的参数（操作系统的缺省设置），则会影响 CPU 在某些区域的操作。

用户程序

你必须自己生成用户程序并下载到 CPU。这个程序中包含处理你的特定的自动化任务所需要的所有功能。用户程序的任务包括以下各项：

- 指定在CPU上暖起动和热起动的条件（例如，带有某个特定值的初始化信号）
- 处理过程数据（例如，二进制信号的逻辑组合、读入并处理模拟信号、为输出指定二进制信号、输出模拟值）
- 指定对中断的响应
- 处理程序的正常运行中的干扰

4.2 用户程序中的块

4.2.1 用户程序中的块

STEP 7 编程软件允许结构化你的用户程序，也就是说可以将程序分解为单个的、自成体系的程序部分。这样做有以下优势：

- 大规模的程序更容易理解
- 可以对单个的程序部分进行标准化
- 程序组织简化
- 程序修改更容易
- 由于可以分别测试各个部分，查错更为简单
- 系统的调试更容易

工业搅拌过程的示例说明了将一个自动化过程分解为单个的任务的优越性。结构化的用户程序的各个部分相应于这些单个的任务，就是大家所知的程序块。

块类型

在 S7 用户程序中有几种不同类型的块可以使用：

块	功能的简要描述	还可参考
组织块（OB）	OB 决定用户程序的结构	组织块和程序结构
系统功能块（SFB） 系统功能（SFC）	SFB 和 SFC 集成在 S7 CPU 中可以让你访问一些重要的系统功能	系统功能块（SFB）和系统功能（SFC）
功能块（FB）	FB 是带有 “ 存储区域 ” 的块，你可以自己编程这个存储区域	功能块（FB）
功能（FC）	FC 中包含经常使用的功能的例行程序	功能（FC）
背景数据块 （背景 DB）	当一个 FB/SFB 被调用时，背景 DB 与该块相关联，它们可在编译过程中自动生成	背景数据块
数据块（DB）	DB 是用于存储用户数据的数据区域，除了指定给一个功能块的数据，还可以定义可以被任何块使用的共享数据	共享数据块（DB）

OB、FB、SFB、FC 和 SFC 都包含部分程序，因此也称作逻辑块。每种块类型所允许的块的数量以及块的长度视 CPU 而定。

4.2.2 组织块和程序结构

组织块（OB）是操作系统和用户程序之间的接口。它们由操作系统调用并控制循环和中断驱动的程序的执行以及可编程控制器如何启动。它们还处理对错误的响应。通过编程组织块，你可以指定 CPU 的反应。

组织块的优先级

组织块决定各个程序部分执行的顺序。一个 OB 的执行可以被另一个 OB 的调用而中断。哪个 OB 可以中断其它 OB，由它的优先级决定。高优先级的 OB 可以中断低优先级的 OB。背景 OB 的优先级最低。

中断的类型和优先级

导致 OB 被调用的事件就是所知的中断。下表显示了 STEP 7 中的中断类型以及分配给它们的组织块的优先级。并非所有列出的组织块和它们的优先级适用于所有的 S7 CPU（见《S7-300 可编程控制器、硬件和安装手册》以及《S7-400、M7-400 可编程控制器模板技术规范参考手册》）。

中断类型	组织块	优先级（缺省）	还可参考
主程序循环	OB1	1	用于循环程序处理的组织块（OB1）
日时钟中断	OB10 至 OB17	2	日时钟中断组织块（OB10-OB17）
时间延迟中断	OB20	3	时间延迟中断组织块（OB20-OB23）
	OB21	4	
	OB22	5	
	OB23	6	
循环中断	OB30	7	循环中断组织块（OB30-OB38）
	OB31	8	
	OB32	9	
	OB33	10	
	OB34	11	
	OB35	12	
	OB36	13	
	OB37	14	
	OB38	15	

中断类型	组织块	优先级（缺省）	还可参考
硬件中断	OB40 OB41 OB42 OB43 OB44 OB45 OB46 OB47	16 17 18 19 20 21 22 23	硬件中断组织块（OB40-OB47）
多处理器中断	OB60 多处理器	25	多处理器-多 CPU 的同步操作
冗余错误	OB70 I/O 冗余错误（只适用于 H 系统） OB72 CPU 冗余错误（只适用于 H 系统）	25 28	故障处理级织块（OB70-OB87/OB121-OB122）
异步的故障	OB80 时间错误 OB81 电源故障 OB82 诊断中断 OB83 插入/移走模板中断 OB84 CPU 硬件故障 OB85 优先级错误 OB86 机架故障 OB87 通讯错误	26 （或者如果异步错误 OB 存在于启动程序中则为 28）	故障处理组织块 块（OB70-OB87/OB121-OB122）
背景循环	OB90	29[1]	背景组织块（OB90）
启动	OB100 暖启动 OB101 热启动 OB102 冷启动	27 27 27	启动组织块 （OB100/OB101/OB102）
同步错误	OB121 编程错误 OB122 访问错误	引起错误的 OB 的优先级	故障处理组织块 （OB70-OB87/OB121-OB122）
1）优先级别 29 相应于优先级 0.29。背景循环的优先级低于自由循环。			

改变优先级

用 STEP 7 可以对中断进行赋值参数，用参数赋值你可以，比如在参数块中不选中中断 OB 或优先级：日时钟中断，时间延迟中断，循环中断和硬件中断。

S7-300 CPU 中组织块的优先级是固定的。

对 S7-400 CPU（以及 CPU318），你可以用 STEP 7 修改以下组织块的优先级：

- OB10至OB47
- OB70至OB72（只适用于H CPU）以及OB81至OB87在RUN模式下。

下列为允许的优先级：

- 优先级2至23相应于OB10至OB47
- 优先级25或28相应于OB70至OB72
- 优先级26或28相应于OB81至OB87

可以将同一个优先级分配给几个 OB。具有相同优先级的 OB 的处理顺序是它们的启动事件出现的顺序。

由同步错误启动的故障 OB 被处理的优先级与错误出现时正在执行的块的优先级一样。

局域数据

生成逻辑块（OB、FC、FB）时可以声明临时局域数据。CPU 上的局域数据区域按优先级划分。

在 S7-400 上，可以使用 STEP 7 在 “ 优先级 ” 参数块中改变每个优先级的局域数据的数量。

OB的启动信息

每个组织块都有 20 个字节局域数据的启动信息，这些信息是一个 OB 被启动时由操作系统提供的。这些启动信息指定了 OB 的启动事件，OB 启动的日期和时间，出现的错误及诊断事件。

例如，OB40 是一个硬件中断 OB，在它的启动信息中包含产生中断的模板地址。

取消选定中断OB

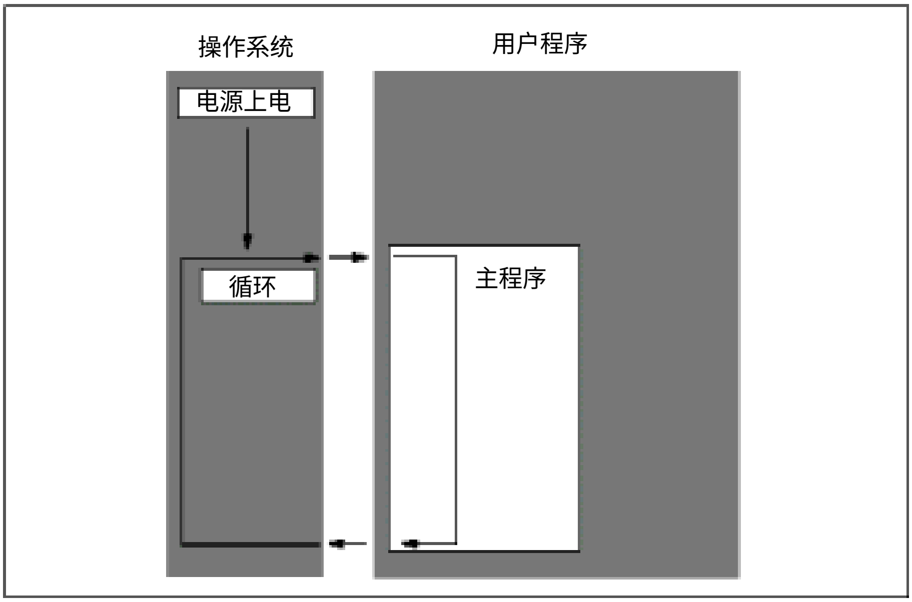
如果将优先级赋值为 0，或分配少于 20 个字节的局域数据给某个优先级，则相应的中断 OB 就被取消选定。对于取消选定的中断 OB 的处理有以下限定：

- 在RUN模式下，它们可以被拷贝或连接到你的用户程序。
- 在STOP模式下，它们可以被拷贝或连接到你的用户程序，但是当CPU进行暖启动时，它们停止这个启动并在诊断缓存区中存入一个输入项。

通过取消选定你不需要的中断 OB，可以增加可供使用的局域数据量，这可以被用来在其它优先级中节省临时数据。

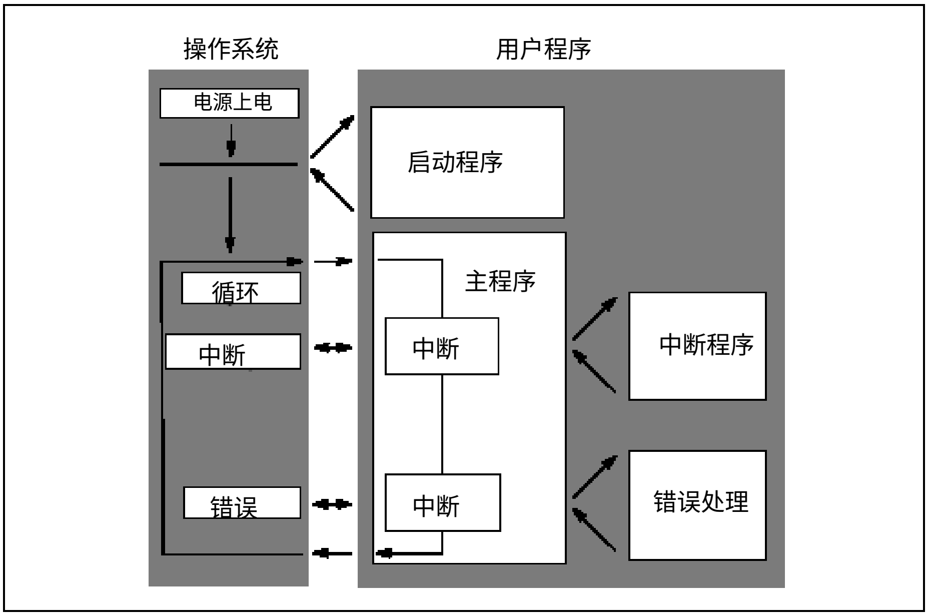
循环程序处理

循环程序处理是可编程控制器上程序执行的“ 普通 ”类型，这意味着操作系统在程序环（循环）中运行，并且在主程序的每一个环中调用一次组织块 OB1。OB1 中的用户程序因此也被循环执行。



事件驱动的程序处理

循环程序处理可以被某些事件中断。如果一个事件出现，当前正在执行的块在语句边界被中断，并且另一个被分配给特定事件的组织块被调用。一旦该组织块执行结束，循环程序将从断点处继续执行。

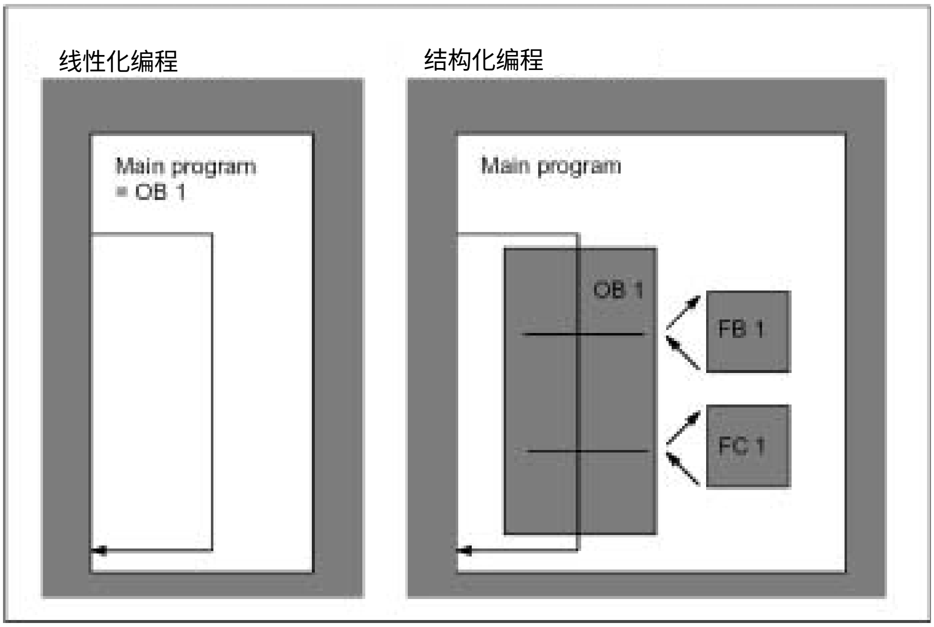


这意味着部分用户程序可以不必循环处理，只在需要时处理。用户程序可以分割为“ 子程序 ”，分布在不同的组织块中。如果用户程序是对一个重要信号的响应，这个信号出现的次数相对较少（例如，用于测量罐中液位的一个限位传感器报警达到了最大上限），当这个信号出现时，要处理的子程序就可以放在一个事件驱动处理的 OB 中。

线性编程与结构化编程

可以将整个用户程序写在 OB1 中（线性化编程）。只有在为 S7-300 编写简单程序并且需要较少存储区域时，才建议使用这种方法。

将复杂的自动化任务分解为能够反映过程的工艺、功能或可以反复使用的小任务时，控制会更加容易。这些任务由相应的程序部分表示，即为所知的块（结构化编程）。

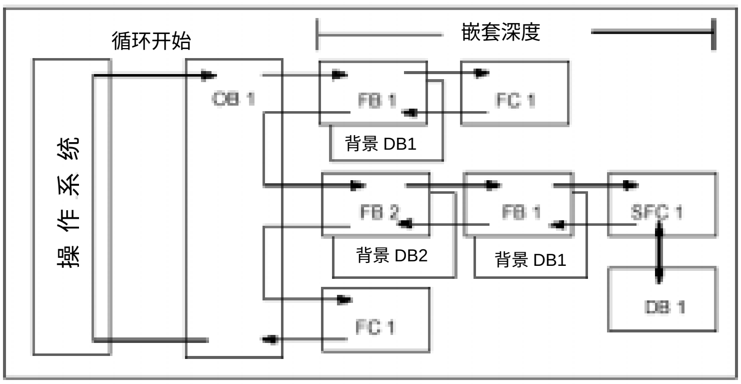


4.2.3 用户程序中调用的分层结构

为使用用户程序工作，组成用户程序的块必须被调用。使用特殊的 STEP 7 指令可以实现块调用，块调用的指令只能在逻辑块中编写和启动。

顺序和嵌套深度

块调用的顺序和嵌套深度即是所谓的调用分层结构。可以嵌套调用的块的数量（嵌套深度）依据特定的 CPU 而定。下图所示为一个循环周期内块调用的顺序和嵌套深度。



创建块的一套顺序：

- 创建块应从上到下，所以你应从最上行的块开始。
- 每个被调用的块应已经存在，即在块的一行中应按从右向左的顺序创建它们。
- 最后创建的块是OB1。

将这些规则用于上图示例的练习，就产生以下块创建的顺序：

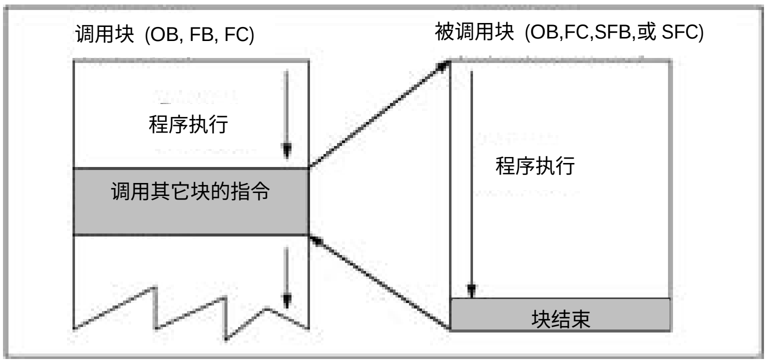
FC1>FB1+背景 DB1>DB1>SFC1>FB2+背景 DB2>OB1

提示

如果嵌套太深（层次太多），局域数据堆栈会溢出（又见局域数据堆栈）。

块调用

下图所示是在一个用户程序中块调用的顺序。程序调用第二个块，这个块的指令则完全被执行。一旦第二个块或者说这个被调用的块执行结束，由于执行调用指令而被中断的块的执行，将从块调用指令后面继续。



在编程一个块之前，必须指定你的程序将使用哪些数据，也就是说，必须声明块的变量。

提示

必须为每个块调用说明 OUT 参数。

提示

当执行冷启动时，操作系统复位 SFB3“TP”的背景。如果你想在一个冷启动之后，初始化这个 SFB 的背景，必须通过 OB100 用 PT=0 调用该 SFB 的相关背景。例如，你可以通过在一个包含该 SFB 背景的块中执行一个初始化例行程序来实现这一步。

4.2.4 块类型和循环程序执行

4.2.4.1 用于循环程序处理的组织块（OB1）

在可编程控制器上循环程序处理是程序执行的“普通”类型。操作系统循环调用 OB1 并用这个调用启动用户程序的循环执行。

循环程序执行的顺序

下表所示为循环程序处理的各个阶段：

步骤	在现有 CPU 中的顺序	在新 CPU 中的顺序（10/98）
1.	操作系统启动循环监控时间	操作系统启动循环监控时间
2.	CPU 读输入模板的输入状态并刷新输入的过程映象表	CPU 从输出的过程映象表写数值到输出模板
3.	CPU 处理用户程序并执行程序中所包含的指令	CPU 读输入模板的输入状态并刷新输入的过程映象表
4.	CPU 从输出的过程映象表写数值到输出模板	CPU 处理用户程序并执行程序中所包含的指令
5.	在循环结束处，操作系统执行所有挂起的任务，例如，下载和删除块，接收和发送全局数据	在循环结束处，操作系统执行所有挂起的任务，如下载和删除块，接收和发送全局数据
6.	最后，CPU 回到循环开始处并重新启动循环监控时间	最后，CPU 回到循环开始处并重新启动循环监控时间

过程映象

所以在循环程序处理过程中，CPU 的过程信号映象是一致的。CPU 并不直接访问 I/O 模板上的输入（I）和输出（Q）地址区域，而是访问一个 CPU 内部的存储区域，该区域中包含输入和输出的映象。

编程循环程序处理

你可以通过使用 STEP 7 在 OB1 中编写你的用户程序以及在 OB1 调用的块中编写程序来编程循环程序处理。

启动程序无错执行一结束，循环程序处理就立即开始。

中断

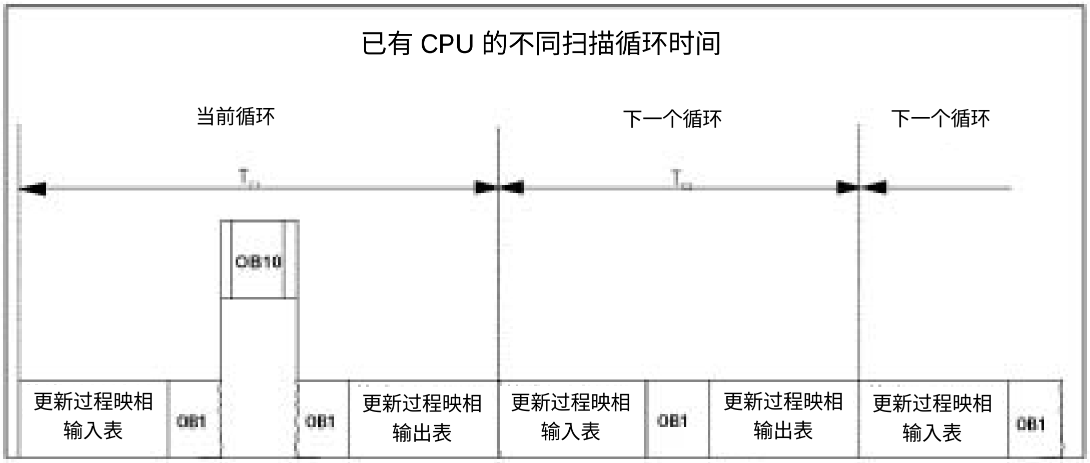
循环程序处理可以被以下事件中断：

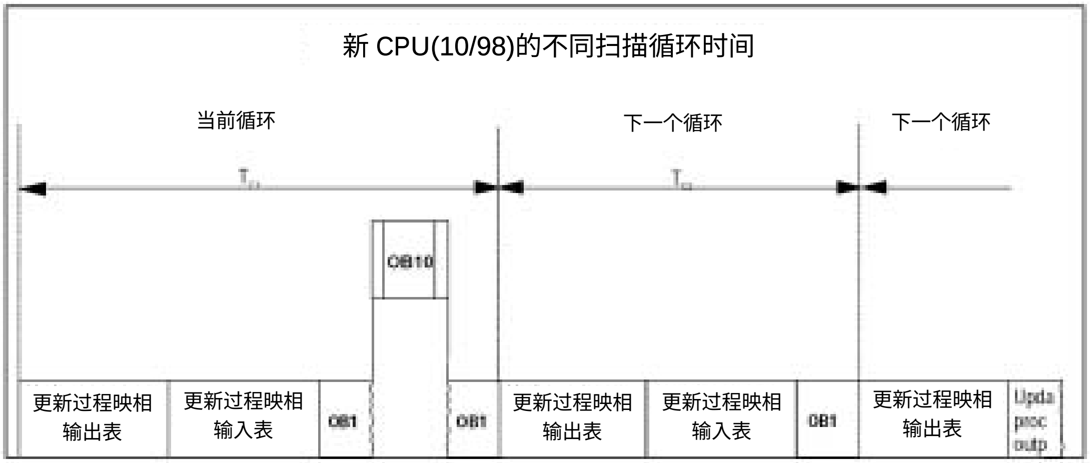
- 一个中断
- STOP命令（模式选择开关，编程器上的菜单选项，SFC46 STP，SFB20 STOP）
- 电源掉电
- 出现故障或编程错误

扫描循环时间

扫描循环时间是操作系统运行循环程序和中断循环的所有程序部分（例如，执行其它组织块）以及系统操作（如，刷新过程映象）所需要的时间。这个时间被监控。

每个循环的循环扫描时间（TC）并不相同。下图所示为已有的 CPU 和新 CPU 之间不同的扫描循环时间（TC1≠TC2）：





在当前循环中，OB1 被日时钟中断。

最大循环时间

使用 STEP 7，可以修改缺省的最大循环时间。如果这个时间超出，CPU 转为 STOP 模式或调用 OB80，在 OB80 中你可以指定 CPU 如何响应这个故障。

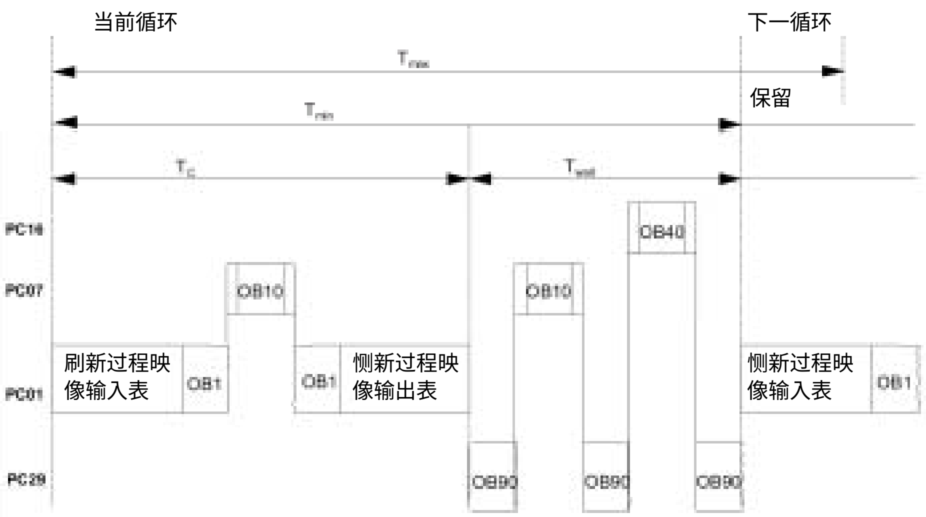
最小循环时间

使用 STEP 7，可以为 S7-400 CPU 和 CPU318 设置最小循环时间。在以下情形下这个功能是有用的：

- 当OB1（主程序扫描）程序执行的启动时间间隔必须一致时，或者
- 如果循环时间太短，过程映象表的刷新没必要过于频繁。

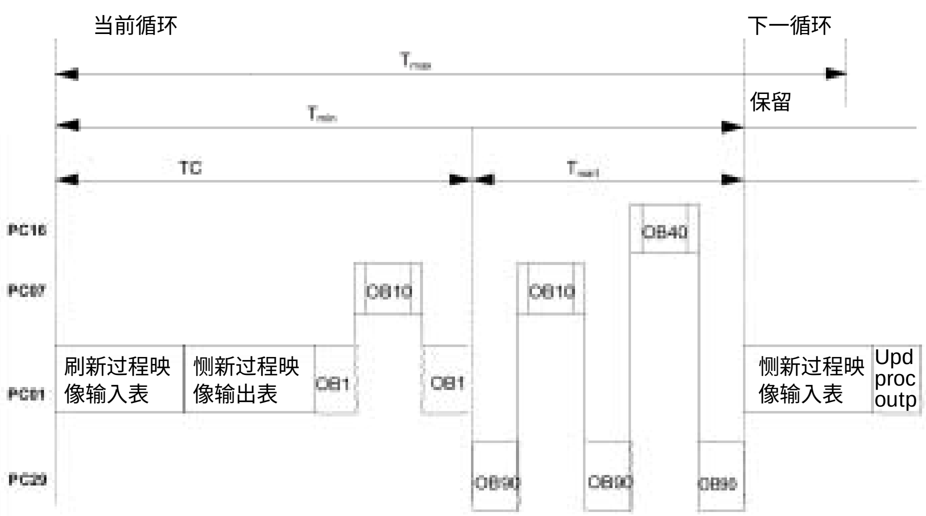
下图所示为已有的 CPU 和新 CPU 中程序处理过程中的循环监控时间的功能。

现有 CPU 的循环监控时间



Tmax = 可设置的最小循环时间
Tmin = 实际扫描循环时间
Tc = Timin 和实际扫描循环时间之差。在这个时期，
Twait = 将出现中断，并可处理背景 OB
PC = 可设置的最大循环时间 PC=优先级

现有 CPU 的循环监控时间



Tmax = 可设置的最小循环时间
Tmin = 实际扫描循环时间
Tc = Timin 和实际扫描循环时间之差。在这个时期，
Twait = 将出现中断，并可处理背景 OB
PC = 可设置的最大循环时间 PC=优先级

刷新过程映象

在 CPU 的循环程序处理过程中，过程映象被自动刷新。对于 S7 400 CPU 以及 CPU 318，你可以刷新过程映象，如果你要：

- 直接访问I/O或者
- 用系统功能SFC26 UPDAT_PI和SFC27 UPDAT_PO在程序的不同点刷新一个或多个过程映象的输入或输出部分。

通讯负载

你可以使用 CPU 参数 “ Scan Cycle Load from Communication（通讯扫描循环负载）”，在给定框架内控制通讯过程的持续时间，一般总为增加扫描循环时间。通讯过程举例包括以 MPI 方式将数据传送至其它 CPU，或使用编程器加载块。

该参数很少影响编程器测试功能。但是，你可以显著增加描述时间。在过程模式中，你可以限制测试功能的时间（只适用于 S7-300）。

参数的工作原理

CPU 操作系统可连续提供整个 CPU 处理能力的通讯组态百分比（时间段技术）。如果通讯时不需要处理功能，可用于其它处理。

设置扫描循环时间

无需其它异步事件，OB1 扫描循环时间可以根据以下公式乘以一个系数进行计算：

$$\frac{100}{100 - \text{通讯扫描循环负载 (\%)}}$$

举例 1（无其它异步事件）：

如果你设定通讯循环加载 50%，OB1 循环时间将加倍。

同时，OB1 扫描循环时间还会受到异步事件的影响（例如硬件中断或循环中断）。据统计，由于扫描循环时间通讯部分的扩展，在 OB1 扫描循环时间内将有更多异步事件发生。这会导致 OB2 扫描循环的额外增加。这种增加取决于每个 OB1 扫描循环时间发生的事件数量以及事件处理时间。

举例 2（考虑其它异步事件）：

对于一个执行时间为 500ms 的 OB1 来说，50%的通讯负载将会导致实际扫描循环时间为 1000ms（CPU 总是有足够的通讯作业要处理）。除此之外，每 100ms 将执行一次处理时间为 20ms 的循环中断，这种循环中断会增加扫描循环 5*20 ms = 100 ms，无通讯负载。即，实际扫描循环时间将为 600 ms。由于一个循环中断也会中断通讯，因此 50%的通讯负载会增加扫描循环时间 10 * 20 ms，即，在这种情况下，实际扫描循环时间为 1200 ms，而不是 1000 ms。

提示

- 检查系统运行时“通讯循环负载”参数的改变效果。
- 在设置最小扫描循环时间时要考虑通讯负载，否则会出现时间错误。

建议

- 如果可能的话，使用缺省值。
- 只有将CPU用于通讯目的，并且用户程序没有时间限制，才能增加该数值。
- 否则，应降低该数值。
- 设定过程模式（只适用于S7-300），限制测试功能所需时间。

4.2.4.2 功能（FC）

功能（FC）属于你自己编程的块。功能是“无存储区”的逻辑块。FC 的临时变量存储在局域数据堆栈中。当 FC 执行结束后，这些数据就丢失了。要将这些数据永久存储，功能也可以使用共享数据块。

由于 FC 没有它自己的存储区，所以你必须为它指定实际参数。不能够为一个 FC 的局域数据分配初始值。

应用程序

一个 FC 包含一个程序部分，当 FC 被不同的逻辑块调用时，这些程序总会被执行。可为以下目的使用功能：

- 为调用块返回一个功能值（例如：数学功能）
- 要执行一个工艺功能（例如：使用位逻辑操作的单控制功能）。

将实际参数赋值给形式参数

形式参数是“ 实际 ”参数的虚名称。当该功能调用时，用实际参数替代形式参数。对于 FC 来说，形式参数总是必须赋给实际参数（例如：将实参“ I 3.6 ”赋值给形参“ Start ”）。在 FC 中使用的参数类型：输入、输出和输入/输出参数存做指针，指向调用 FC 的逻辑块的实际参数。

4.2.4.3 功能块（FB）

功能块（FB）属于用户自己编程的块。。功能块是具有“ 存储功能 ”的块。用数据块作为功能块的存储器（背景数据块）。传递给 FB 的参数和静态变量存在背景数据块中。临时变量存在本地数据堆栈中。

当 FB 执行结束时，存在背景 DB 中的数据不会丢失。可是，当 FB 的执行结束时、存在本地数据堆栈中的数据将丢失。

提示

为避免工作在 FB 时出现错误，当在附录中选择参数时，去读允许的数据类型。

应用程序

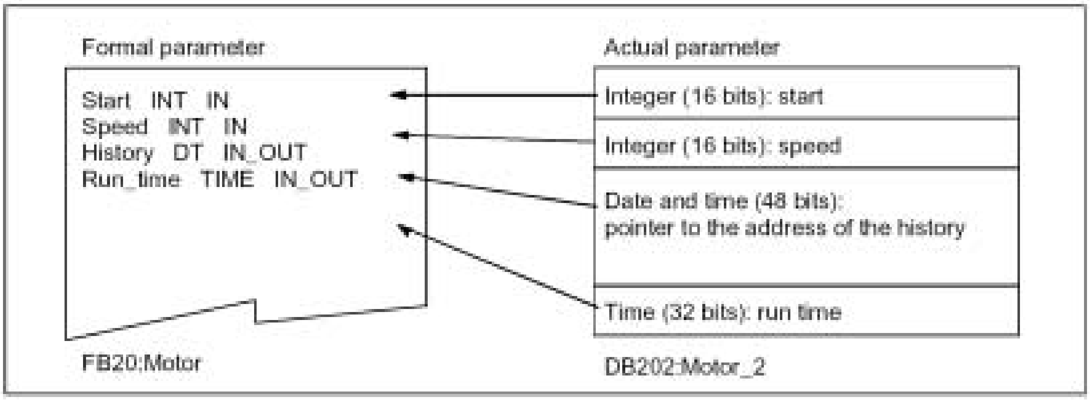
FB 中所含的程序总是当不同的逻辑块调用该 FB 时执行。功能块使得对于经常使用的功能、复杂功能的编程变得容易。

功能块和背景数据块

每次功能块的调用都将赋给一个背景数据块，用于传递参数。

多次调用某一 FB，可以有多个背景，用户可以用一个 FB 控制多台设备。例如，一个用于电机控制的 FB，可以通过对每个不同的电机，使用不用的背景数据集，来控制多台电机。每台电机的数据（例如：转速、爬升、累积运行时间等），可存在一个或多个背景 DB 中。

下图所示为 FB 的形参用实参赋值后并存在背景 DB 中的情况。



数据类型FB的变量

如果用户程序是结构化的，则在 FB 中又调用了另一个已经存在的功能块，用户可在调用 FB 的变量定义表中将被调 FB 作为数据类型 FB 的静态变量。该项技巧允许用户嵌套变量，并将背景数据压缩在一个背景数据块（多重背景）中。

将实际参数赋值给形式参数

在 STEP 7 中，对于 FB 通常不是必须将实际参数赋值给形参。可是，下列情况除外，对以下形参必须赋实参：

- 对于复杂数据类型（如：字符串（STRING），数组（ARRAY）或日期与时间（DATE__AND__TIME）的输入/输出类型参数。
- 对于所有的参数类型（例如，定时器（TIMER）、计数器（COUNTER）或指针（POINTER）。

对于下列情况，STEP 7 会将实际参数赋值给 FB 的形式参数：

- 当用户在调用语句中定义了实际参数时：FB的指令用所提供的实参。
- 当用户在调用语句中没有定义实际参数：FB的指令用存在背景DB中的值。

下表所示为哪些 FB 的变量必须赋实参。

变　量		数据类型	
	基本数据类型	复杂数据类型	参数类型
输入	无实参要求	无实参要求	有实参要求
输出	无实参要求	无实参要求	有实参要求
输入/输出	无实参要求	有实参要求	--

给形式参数赋初值

在 FB 的定义表中，用户可给形式参数赋初值。这些值将写入与 FB 相关的背景 DB 中。

如果用户在调用语句中，没有给形参赋实参，则 STEP 7 将使用存在背景 DB 中的值。这些值也可作为初值输入到 FB 的变量定义表中。

下表所示为哪些变量可以赋初值。由于临时数据在该块执行完后将丢失，所以用户不能给它们赋任何值。

变 量		数据类型	
	基本数据类型	复杂数据类型	参数类型
输入	允许有初值	允许有初值	--
输出	允许有初值	允许有初值	--
输入/输出	允许有初值	--	--
静态	允许有初值	允许有初值	--
临时	--	--	--

4.2.4.4 背景数据块

每次功能块的调用都将赋给一个背景数据块，用于传递参数。FB 的实际参数和静态数据存在背景 DB 中。在 FB 中定义的变量，决定背景数据块的结构。背景意味着一次功能块调用。例如，如果在 S7 用户程序中某个功能块被调用了五次，则该块有五个背景。

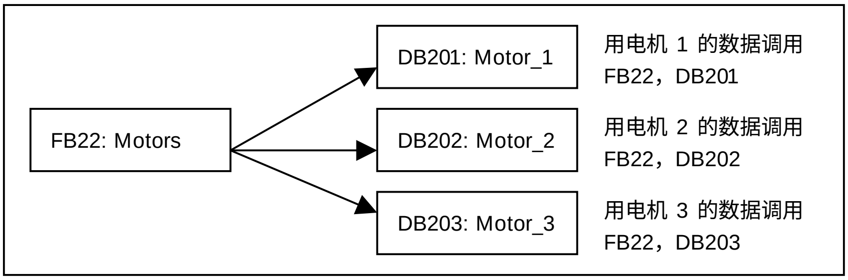
生成一个背景DB

在用户生成一个背景数据块之前，相应的 FB 必须已经存在。当用户生成背景数据块时，必须指定所属 FB 的序号。

每次不同的背景用一个背景DB

如果用户将多个背景数据块分配给某个控制电机的功能块（FB），则用户可用该 FB 去控制多个不同的电机。

描述电机的各项数据（例如：转速、升速时间、整个操作时间），存在不同的数据块中，当 FB 调用时，相应的 DB 决定哪个电机被控制。根据该项技巧，控制多台电机只需一个功能块（参看下图）。

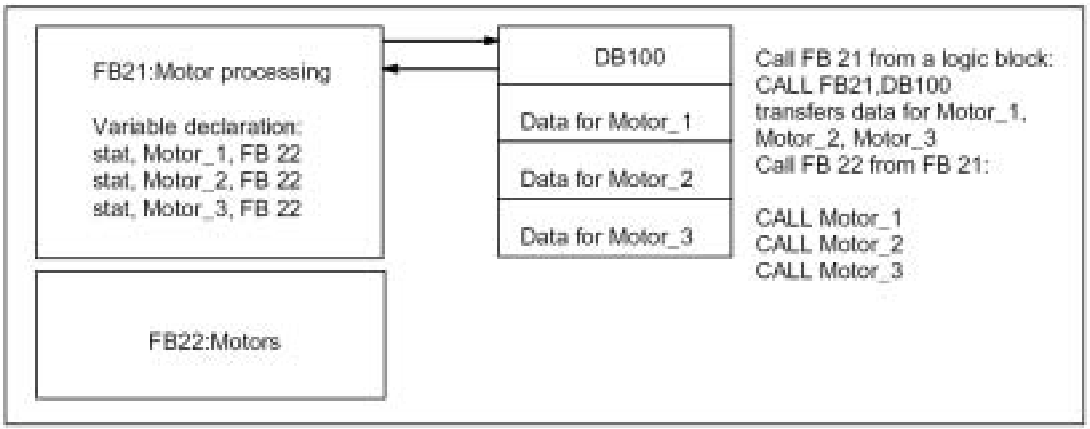


一个背景DB用于某个FB的多次背景（多重背景）

用户也可以将多个电机的背景数据同时传递到一个背景 DB。为此，用户必须增加一个 FB 来管理电机控制器的多次调用，并且，在调用 FB 的定义表中用数据类型 FB 的静态变量定义每个背景。

如果用户只用一个背景 DB 存放某个 FB 的多次背景，则节约了存储空间，并能最优地使用数据块。

在下图中，调用 FB 是 FB21 “ 电机控制 ”，变量为数据类型 FB22，不同的背景用 “ Motor-1 ”、 “ Motor-2 ” 和 “ Motor-3 ” 标识。



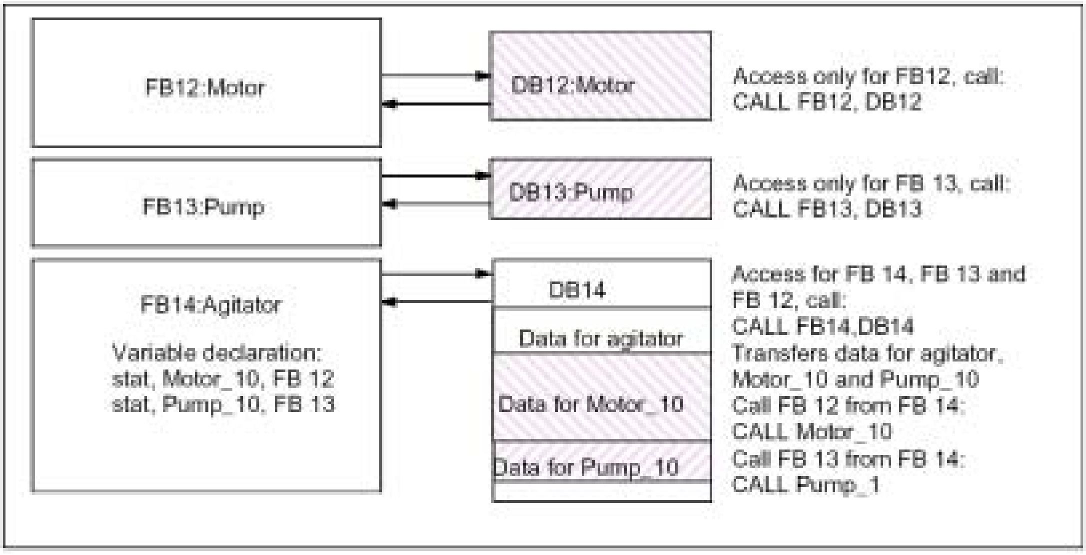
在这个例子中，FB22 不需要自己的背景数据块，因为它的背景数据存在调用 FB 的背景数据块中。

一个背景DB用于不同FB的多次背景（多重背景）

在一个功能块中，用户可以调用其它已经存在的 FB 的背景。为此，用户可以将所需的背景数据赋值到调用 FB 的背景数据块中，这就意味着，在这种情况中，用户不需要为被调 FB 增加任何数据块。

为了将这些多重背景在一个背景数据块中实现，用户必须在调用功能块的定义表部分，为每次独立的背景定义一个被调功能块数据类型的静态变量。在功能块内部的调用，则不再需要背景数据块，只需要变量的符号名。

下图示例中，赋值的背景数据存在一个共同的背景 DB 中。



4.2.4.5 共享数据块（DB）

与逻辑块不同，在数据块中没有 STEP 7 的指令。它们用于存放用户数据，换句话说，数据块中存放用户程序工作时所需的变量数据。共享数据块用于存放所有其它块都可以访问的用户数据。

DB 的大小可以不同。关于所允许的最大尺寸，请参考用户所用 CPU 的描述。

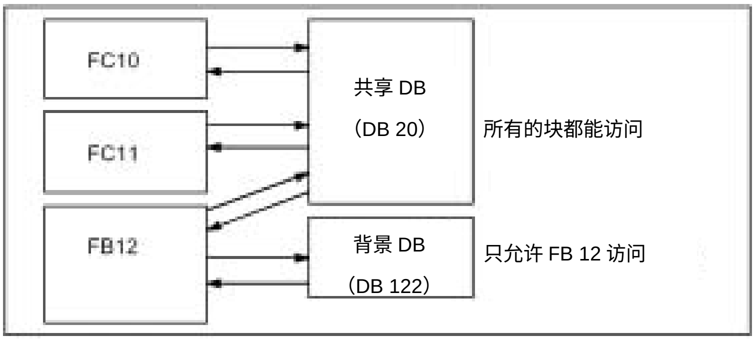
用户可以用任意方式来建立数据块的结构，以适合其不同的需求。

在用户程序中的共享数据块

如果某个逻辑块（FC，FB 或 OB）被调用，则它可以临时占用局域数据区的空间（L 堆栈）。除了这个局域数据区，逻辑块还可以打开一个 DB 形式的存储区。与局域数据区中的数据不同，在 DB 中的数据当 DB 关闭时，换句话说，当相应的逻辑块结束时，不会不删除。

每个 FB、FC 或 OB 可从共享 DB 中读取数据，或将数据写入共享 DB。当该 DB 退出时，这些数据保持在 DB 中。

一个共享 DB 和一个背景 DB 可同时打开。下图所示为访问数据块的不同方法。



4.2.4.6 系统功能块（SFB）和系统功能（SFC）

已经编好程序的块

用户不需要每个功能都自己编程。S7 CPU 为用户提供了一些已经编好程序的块，这些块可在用户程序中进行调用。

在系统功能块和系统功能中的参考帮助中可找到进一步的信息（跳转到语言描述，和有关块的在线帮助以及系统属性）。

系统功能块

系统功能块（SFB）是集成在 S7 CPU 中的功能块。SFB 作为操作系统的一部分，不占用户程序空间。与 FB 相同，SFB 也是“具有存储能力”的块。用户也必须为 SFB 生成背景数据块，并将其下载到 CPU 中作为用户程序的一部分。

S7 CPU 提供下列 SFB：

- 通过组态连接用于通讯目的。
- 集成的特殊功能（例如：CPU 312 IFM 和CPU 314I FM上的 SFB 29 “HS_COUNT”）。

系统功能

系统功能是集成在 S7 CPU 中预先编好程序并通过测试的功能。可在用户程序中调用 SFC。SFC 属于操作系统的一部分，而不算做用户程序的一部分。与 FC 相同，SFC 是“不具有存储能力”的块。

S7 CPU 提供以下功能：

- 复制及块功能
- 检查程序
- 处理时钟和在线运行仪表
- 传递数据集
- 在多CPU状态中将事件从一个CPU传到所有其它的CPU中
- 处理日期时间中断和延时中断
- 处理同步错误、中断错误和异步错误
- 有关静态和动态系统数据的信息，例如：诊断
- 过程映象刷新和位域处理
- 寻址模板
- 分布式I/O
- 全局数据通讯
- 非组态连接的通讯
- 生成块相关信息

其它信息

有关 SFB 和 SFC 更详细的信息，请参考《S7-300 和 S7-400 系统软件，系统和标准功能》参考手册，《S7-300 可编程序控制器，硬件和安装手册》以及《S7-400，M7-400 可编程序控制器模板特性参考手册》中关于 SFB 和 SFC 的解答。

4.2.5 用于中断程序处理的组织块

4.2.5.1 用于中断程序处理的组织块

根据所提供的中断 OB，S7 CPU 允许如下中断：

- 可按照一定的时间或间隔来执行某一部分程序（时间中断）。
- 用户程序可对过程中的外部信号进行响应。

循环执行的用户程序不需要查询是否有中断事件发生。如果有中断请求，操作系统确保执行在相应的中断 OB 中的程序，这就使得可编程序逻辑控制器对于中断，可以编程进行响应。

中断类型与应用

下表所示为如何使用不同类型的中断。

中断类型	中断 OB	应用举例
日期时间中断	OB10 到 OB17	在一个班结束时，计算一下混合流量控制的整个流量。
延时中断	OB20 到 OB23	控制风扇：当电机关掉之后 20 秒时，风扇必须运行。
循环中断	OB30 到 OB38	对于闭环控制系统，信号的定时采样。
硬件中断	OB40 到 OB47	当罐的最高上限达到时、产生报警信号

4.2.5.2 日时钟中断组织块（OB10-OB17）

S7 CPU 提供日期时间中断 OB，这些 OB 在特定的日期和时间或以一定间隔由操作系统调用执行。

日期时间中断可按如下方式触发：

- 在某特定时间（用绝对形式定义日期时间）执行一次。
- 从特定的时间开始并按中断应重复的间隔（例如：每分钟、每小时、每天）周期地执行。

日期时间中断规则

日期时间中断只有当该中断设置了参数，并且在相应的组织块中有用户程序存在时才能被执行。如果与上述情况不符，则操作系统会诊断缓存器中输入一个错误信息，并执行异步错误处理（OB80，请参看错误处理组织块（OB70 到 OB87/OB121 到 OB122））。

周期的日期时间中断必须对应一个实际日期。从一月三十一日开始每月重复 OB10 是不可能的。在这种情况下，该 OB 将只在有 31 天的那个月起动。

日期时间中断在 PLC 起动时（暖起动或热起动）激活，而且，只能在 PLC 起动过程结束之后，才能执行。

在参数设置时，没有选中的日期时间中断不能起动。CPU 认为这是一个编程错误，将进入到停机状态。

暖起动之后，日期时间中断必须重新设置（例如，在 PLC 起动程序中用 SFC30 ACT__TINT）。

起动日期时间中断

为了让 CPU 起动日期时间中断，用户必须首先设置日期时间中断，然后再激活它。起动该中断有以下三种方法：

- 通过STEP 7中设置相应的参数（“ 日期时间中断 ” 参数块），实现日期时间中断的自动起动。
- 在用户程序中用SFC28 SET_TINT和SFC 30 ACT_TINT，设置并激活日期时间中断。
- 用STEP 7的参数设置日期时间中断，在用户程序中用SFC 30 ACT_TINT激活日期时间中断。

查询日期时间中断

要想查询设置了哪些日期时间中断，以及这些中断什么时间发生，用户可选用下列方法之一：

- 调用SFC 31 QRY_TINT。
- 请求系统状态表的“ 中断状态 ” 表。

终止日期时间中断

用 SFC 29 CAN_TINT，用户可以取消那些还没有执行的日期时间中断。用 SFC 28 SET_TINT 可以重新设置那些被终止的日期时间中断，并可用 SF 以 30 ACT_TINT 重新激活它们。

日期时间中断OB

所有八个日期时间中断 OB 具有相同的预置优先级（2），因此，它们的优先级是按起动事件发生的顺序进行处理的。可是，用户可以通过选择适当的参数来改变优先级。

改变设置时间

用户可以用如下方法改变中断的日期时间设置：

- 用主时钟同步主站和从站的时间。
- 在用户程序中可以调用SFC0 SET_CLK设置一个新的时间。

改变时间的响应

下表所示为日期时间中断在时间已经改变之后，是如何反应的。

如果…	则…
如果时间向前挪动，并且有一个或多个日期时间中断已经跳过	起动 OB80，那些跳过的日期时间中断输入到 OB80 的起动信息中
用户没有终止在 OB80 中跳过的日期时间中断	跳过的那些日期时间中断，将不再执行
用户还没有终止在 OB80 中跳过的日期时间中断	第一个跳过的日期时间中断执行。其它的跳过的日期时间中断将忽略
将时间设置向后挪动，日期时间中断的起动事件会再次发生	日期时间中断的执行不再重复

4.2.5.3 时间延迟中断组织块（OB20-OB23）

S7 CPU 提供延时 OB 用于在用户程序中编写延时执行的程序。

延时中断规则

延时中断只有当相应的组织块在 CPU 程序中存在时，才能执行。如果与上述情况不符，则操作系统会诊断缓存器中输入一个错误信息，并执行异步错误处理（OB80，请参看错误处理组织块（OB70 到 OB87/OB121 到 OB122））。

在参数设置时，如果没选延时中断 OB，则该 OB 不能起动。CPU 认为这是一个编程错误，将进入到停机状态。

当在 SFC 32 SRT__DINT 中设定的延时时间达到时，触发延时中断。

起动延时中断

为了起动延时中断，用户必须在 SFC 32 中设定延时时间，当相应的时间延迟达到时，该中断 OB 被调用。请参考《S7-300 可编程序控制器，硬件及安装手册》和《S7-400，M7-400 可编程序控制器模板规范参考手册》中有关最大允许的延迟时间长度。

延时中断OB的优先级

延时中断 OB 优先级的缺省设置值为 3 至 6 级。用户可以设置参数改变优先级。

4.2.5.4 循环中断组织块（OB30-OB38）

S7 CPU 提供循环中断 OB，可用于按一定间隔中断循环程序的执行。

循环中断按间隔触发。间隔的时间是从 STOP 状态到 RUN 时开始计算。

循环中断的规则

当用户定义时间间隔时，必须确保在两次循环中断之间的时间隔中，有足够的时间处理循环中断自己的服务程序。

如果用户在设置参数时，没有选循环中断 OB，它们将不再起动。CPU 认为这是一个编程错误，将进入到停机状态。

起动循环中断

为了起动循环中断、用户必须在 STEP 7 中的循环中断参数块里定义时间间隔。时间间隔必须是 1ms 基本时钟率的整数倍。

时间间隔=n×基本时钟率 1ms

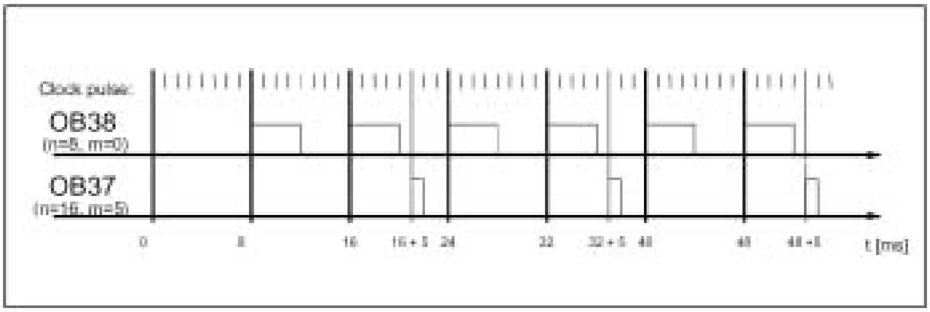
一共有九个循环中断 OB，每个 OB 都有其缺省的时间间隔（请参看下表）。为相应的循环中断 OB 装载后，其对应的缺省时间间隔变为有效。然而，用户可以设置参数改变缺省值。请参阅《S7-300 可编程序控制器，硬件和安装手册》及《S7-400，M7-400 可编程序控制器模板规范参考手册》中有关上限的内容。

循环中断中的相位偏移

为避免不同的循环中断 OB 在同时开始请求中断，可能造成时间错误（循环时间超过），用户可以定义一个相位偏移。相位偏移确保当循环中断的时间间隔达到时，再做一定的延时。

相移=m×基本时钟率（0≤m<n）

下图所示为与无相移循环中断（OB38）相比，相移循环中断 OB（OB37）的执行情况。



循环中断OB的优先级

下表所示为循环中断OB缺省的时间间隔和优先级。用户可以设置参数来改变时间间隔和优先级。

循环中断	时间间隔（ms）	优先级
OB30	5000	7
OB31	2000	8
OB32	1000	9
OB33	500	10
OB34	200	11
OB35	100	12
OB36	50	13
OB37	20	14
OB38	10	15

4.2.5.5 硬件中断组织块（OB40-OB47）

S7 CPU 提供有硬件中断 OB，用于对模板（例如，信号模板（SM），通讯处理器（CP），功能模板（FM））上的信号变化进行响应。通过 STEP 7，用户可以决定从可组态的数字量或模拟量模板来的哪些信号可以起动 OB。对于 CP 和 FM，可能使用在对话框中设置相应的参数来起动 OB。

当具有中断能力的信号模板上，有一个允许进行中断的信号从过程传到 CPU 时，或者当 CPU 的功能模板产生一个中断信号时，将触发硬件中断。

硬件中断的规则

硬件中断只有当 CPU 的程序中存在相应的组织块时，才能执行。如果与上述情况不符，则操作系统会诊断缓存器中输入一个错误信息，并执行异步错误处理（OB80，请参看错误处理组织块（OB70 到 OB87/OB121 到 OB122））。

如果用户在参数设置中没有选中硬件中断 OB，则它们不能起动。CPU 认为这是一个编程错误，将进入到停机状态。

具有硬件中断能力的信号模板的参数设置

具有硬件中断能力信号模板的每个通道都可以触发一个硬件中断。为此，用户通过 STEP 7 必须给具有硬件中断能力的信号模板设置如下参数集：

- 用什么触发一个硬件中断
- 哪种硬件中断OB将被执行缺省设置（OB40用于执行所有的硬件中断）

用户通过 STEP 7，可以使用功能块激活硬件中断的生成。在这些功能模块的参数设置对话框中，设置保持的参数。

硬件中断OB的优先级

硬件中断 OB 的缺省优先级为 16 到 23。用户可以设置参数改变优先级。

4.2.5.6 启动组织块（OB100/OB101/OB102）

起动类型

起动特性有三种不同的类型：

- 热起动（在S7-300和S7-400H中没有）
- 暖起动
- 冷起动

下表所示为：对应各种起动类型，操作系统调用不同的 OB。

起动类型	相关 OB
热 起 动	OB101
暖 起 动	OB100
冷 起 动	OB102

起动OB的起动事件

CPU 当下列事件发生后，执行起动功能：

- 电源上电后
- 用户将CPU的状态选择开关从“ STOP ”拔到“ RUN/RUN-P ”后
- 从通讯功能来的请求后
- 多CPU方式同步之后
- H系统中连接后（只适用于备用CPU上）

根据起动事件、所使用的 CPU 及其设置参数，调用相应的起动 OB（OB100、OB101 或 OB102）。

启动程序

用户可以通过在暖起动的组织块 OB100、热起动的组织块 OB101 或冷起动的组织块 OB102 中编写程序，来设定其 CPU 起动的起动条件（运行时的初值，I/O 模板的起动值）。

起动程序没有长度限制，也没有时间限制，因为循环监视还没激活。在起动程序中，不能执行时间中断或硬件中断程序。在起动过程中，所有数字量输出的信号状态都为“ 0 ”。

手动起动后的起动类型

S7-300 CPU 只允许手动暖起动或冷起动（只有 CPU 318-2）。

对于某些 S7-400 CPU，如果用户通过 STEP 7 的参数设置允许手动起动，则用户可以使用状态选择开关和起动类型开关（CRST/WRST），进行手动起动。手动暖起动可以不用设定参数。

自动起动后的起动类型

对于 S7-300 CPU，电源上电后，只允许暖起动。

对于 S7-400 CPU，用户可定义电源上电后，是自动暖起动还是自动热起动。

过程映象清零

当 S7-400 CPU 起动时，将执行保留的循环，并作为缺省设置，清零输出过程映象表。如果用户希望在起动之后，继续在用户程序中使用旧值，也可以不将过程映象清零。

模板存在/类型监视

当用户进行组态表的参数设置时，可以决定是否需要在组态表中检查模板是否存在，以及模板类型与起动前是否匹配。

如果模板检查激活，CPU 发现组态表与实际组态不相符时，CPU 将不起动。

监视时间

为确保可编程序控制器起动时没有错误，用户可以选择以下监视时间：

- 模板传递参数的最大允许时间
- 上电后，模板准备好用于操作的信号所允许的最大时间
- 对于S7-400 CPU，热起动期间中断所允许的最大时间

一旦监视时间达到，CPU 将进入到停机状态或只能做暖起动。

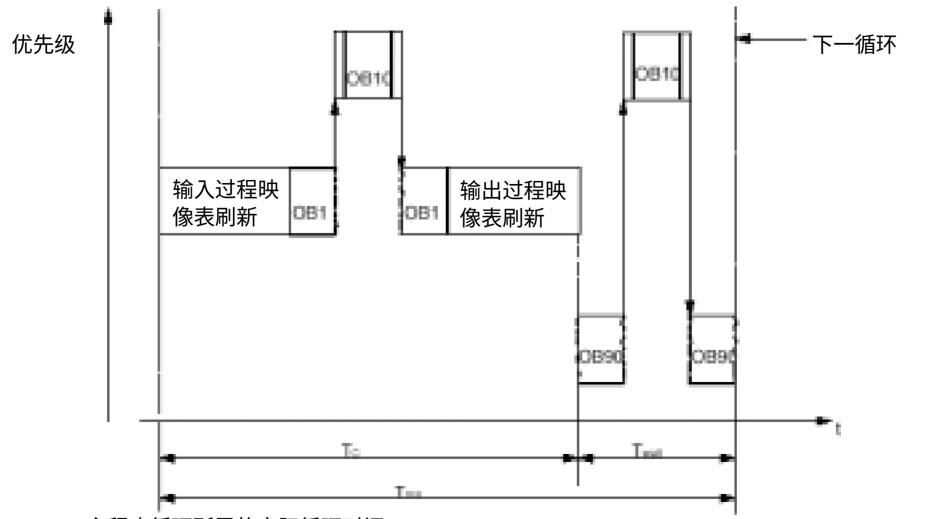
4.2.5.7 背景组织块（OB90）

如果用户用 STEP 7 定义最小的扫描循环时间，且该时间比实际的扫描循环时间长，则 CPU 在循环程序结束时，还有处理时间。该时间用于执行背景 OB。如果用户的 CPU 中没有 OB90，则 CPU 等待，直到定义的最小扫描循环时间达到为止。因此，对于那些对运行时间要求不高的过程，用户可以用 OB90，而避免等待时间。

背景OB的优先级

背景 OB 的优先级为 29，对应的优先级 0.29。因此，该 OB 的优先级最低。其优先级不能通过参数设置进行修改。

下图所示为，在CPU中处理的背景循环、主程序循环和OB10。



Tc = 主程序循环所需的实际循环时间
Tmin = 用 STEP 7 设定的最小循环时间
Twait = 下一个循环开始之前所剩的循环时间

OB90的编程

由于 OB90 的运行时间不受 CPU 操作系统的监视，因此，用户可以在 OB90 中编写程序的长度不受限制。为确保在背景程序中的数据具有一致性，在编程时注意以下问题：

- OB90的清零事件（参看《S7-300和S7-400的系统软件、系统和标准功能》参考手册）
- 过程映象的刷新与OB90不同步

4.2.5.8 故障处理组织块（OB70-OB87/OB121-OB122）

错误类型

可被 S7 CPU 检测到，并且，用户可以通过组织块对其进行响应的错误，可分为两个基本类型：

- 同步错误：这些错误可能出现在用户程序的某一部分中。该类错误发生在执行某特定指令过程中。如果没有装载相应的同步错误OB，当错误发生时，CPU进入到STOP状态。

- 异步错误：这些错误不会出现在用户程序的执行过程中。该类错误是优先级错误、可编程序控制器故障（例如：模板损坏）或冗余错误。如果没有相应异步错误OB，当错误发生时，CPU进入到“ STOP ”状态。
- （例外：OB70，OB72，OB81）。

下表所示为可能出现的错误类型，按错误处理 OB 进行分类。

异步错误/冗余错误	同步错误
OB70 I/O 冗余错误（只适用于 H CPU）	OB121 编程错误（例如没有装入 DB）
OB72 CPU 冗余错误（只适用于 H CPU，例如，一个 CPU 发生故障）	OB122 I/O 存取错误（例如，存取一个并不存在的信号模块）
OB73 通讯冗余错误（只适用于 H CPU，例如，一个冗余 S7 连接冗余损失）	
OB80 计时错误（例如，超过扫描循环时间）	
OB81 电源故障（例如，电池故障）	
OB82 诊断中断（例如，输入模块短路）	
OB83 拆除/插入中断（例如，拆除一个输入模块）	
OB84 CPU 硬件故障（ MPI 网络接口故障）	
OB85 优先级错误（例如没有装入 OB）	
OB86 机架故障	
OB87 通讯错误（例如，一个不正确的全局数据通讯信息框 ID）	

同步错误OB

同步错误发生在执行某一特定指令的过程中。当这些错误出现时，操作系统在 I 堆栈做一个输入记录，并调用同步错误 OB。

该类型错误 OB 的调用，是由于在程序中执行了一个同步错误的结果，其优先级与检测到错误的块一致。因此，OB121 和 OB122 可以访问中断发生时的累加器与其它寄存器中的内容。用户可利用这些值，对错误条件进行响应，然后，返回处理用户程序（例如，如果访问某个模拟输入模板时，有个访问错误，则用户可以在 OB122 中用 SFC44 RPL_VAL 定义一个替代值）。可是，该错误 OB 的本地数据，在 L 堆栈中的这个优先级中，额外占用一个空间。对于 S7-400 CPU，一个同步错误 OB 可以起动另一个同步错误 OB。对 S7-300 CPU，这个功能不可能。

异步错误OB

如果 CPU 的操作系统检测到一个异步错误时，将起动相应的错误 OB（OB70 到 OB73 和 OB80 到 OB87）。该类异步错误 OB 具有最高等级的优先级，如果所有的异步错误 OB 的优先级相同，则它们不能被其它 OB 中断。如果同时有多个相同优先级的异步错误 OB 出现，它们将按其出现的顺序进行处理。

屏蔽起动事件

利用系统功能（SFC），用户可以屏蔽、延迟或禁止各种 OB 的起动事件。有关这些 SFC 和组织块的详细信息，请参考《S7-300 和 S7-400 系统软件，系统和标准功能》参考手册。

提示

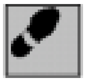
如果用户希望忽略中断，更有效的方法不是禁止它们，而是下载一个空的 OB（内容只有 BE）。

---

5 启动和操作

5.1 启动 STEP 7

5.1.1 启动 STEP 7

 启动 Windows 以后，你就会发现一个 SIMATIC Manager（SIMATIC 管理器）的图标，这个图标就是启动 STEP 7 的接口。

快速启动 STEP 7 的方法：将光标选中 SIMATIC Manager 这个图标，并双击。打开 SIMATIC 管理器窗口。从这里，你可以访问你所安装的标准模块和选择模块的所有功能。

启动 STEP 7 的另一方式：在 Windows 95/98/NT 的任务栏中选中“ Start ”键，而后进入“ Simatic ”。

注意

你可以从 STEP 7 窗口的用户引导 Windows 打开程序在线提示，得到更多的有关标准窗口的操作及选项的信息。

SIMATIC 管理器

SIMATIC 管理器用于基本的组态和编程。SIMATIC 管理器具有下列功能：

- 建立项目
- 硬件组态及参数设定
- 组态硬件网络
- 编写程序
- 编辑、调试程序

对各种功能的访问都设计成直观、易学的方式。

可以使用 SIMATIC 管理器在下列方式工作：

- 离线方式，不与可编程控制器相联
- 在线方式，与可编程控制器相联

注意相应的安全提示。

下步如何进行呢?

以 " Projects " 形式建立自动化作业标题。在操作之前，如果先理解下列内容，你就会感到这个操作变得非常简单：

- 用户接口
- 基本操作步骤
- 在线帮助

5.1.2 启动 STEP 7 并带有预置启动参数

使用 STEP 7 V5.0 以上版本，你可以在 SIMATIC 管理器中生成几个符号，并在调用顺序上指定启动参数。为此，SIMATIC 管理器选中这些参数描述的对象，快速双击，立即进入相应的 Project。

执行文件 S7tgtopx.exe，可以指定下列起动参数：

/e <完整的物理 Project 路经>

/o <对象的逻辑路径>

/h <对象 ID> /on 或/off

下列描述为建立相应参数最简捷的方法。

用复制和粘贴方式建立参数

按如下进行：

1. 建立访问文件S7tgtopx.exe的路径。
2. 打开属性对话框。
3. 选择“ Link ”选项卡。如下扩展“ Target ”下的输入。
4. 在SIMATIC管理器选择相应的对象。
5. 用“ CTRL+C ”复合键，复制这个对象到文件夹。
6. 将光标移“ Link ”选项卡中“ Target ”的末端。
7. 用“ CTRL+V ”复合键，粘贴文件夹中的内容。
8. 用“ OK ”关闭对话框。

参数举例：

/e F:\SIEMENS\STEP7\S7proj\MyConfig\MyConfig.s7p

/o ”1,8:MyConfig\SIMATIC 400（1）\CPU416-1\S7-Program（1）\Blocks\FB1_

/h T00112001;129;T00116001;1;T00116101;16e

注意 Project 路径的结构

在文件系统中 Project 路径是一个物理路径。不支持 UNC Notation，如：F:\SIEMENS\STEP7\S7proj\MyConfig\MyConfig.s7p

完整的逻辑路径如下：

[View ID, online ID]:project name\{object name\}*\ 对象名

例如：/o 1.8:MyConfig\SIMATIC 400（1）\CPU416-1\S7-Program（1）\Blocks\FB1

注意逻辑路径的结构

只能用复制和粘贴功能，建立完整的逻辑路径和对象 ID。

但是，用户也可以访问这些路径。如前例所述：

/o "MyConfig\SIMATIC 400（1）\CPU416-1\S7-Program（1）\Blocks\FB1”。通过增加\on 或\off，用户可以指明路径是否在在线窗口或离线窗口有效。如果仅做复制和粘贴则无需做上述工作。

重要：如果路径中有空格，应用引号代替这些空格。

5.1.3 访问帮助功能

在线帮助

在线帮助系统提供给用户有效快速的信息，无需查阅手册，在线帮助具有如下信息方式：

- Contents：显示帮助信息的号码
- Context-Sensitive Help（F1 key）：首先用鼠标选中或在对话框或窗口选择某一对象，而后用F1键得到相应帮助信息。
- Introduction：对某种功能的使用、主要特性及功能范围做一个简要说明。
- Getting Started：概述启动某功能的基本步骤。
- Using Help：在在线帮助下，对查找特殊信息的方法提供描述。
- About：提供有关当前版本的信息。

通过帮助菜单可访问任何窗口的实时对话框。

访问在线帮助功能

你可以用下列方法之一访问在线帮助：

- 在菜单栏选中帮助菜单，再从中选择相应指令。
- 在某对话框单击“ Help ”键，则可以显示这个对话框的帮助。
- 将光标移到某个你希望得到帮助的窗口或对话框，而后按下F1键或选择菜单命令Help > Context-Sensitive Help。
- 在窗口内使用问号键。

在线帮助功能访问的后三种方式即所熟悉的上下文相关帮助。

访问快速帮助功能

你只要将光标移到需要帮助的工具栏的某个键上，保持片刻，则显示出快速帮助信息。

改变字符的大小

使用 Windows 的菜单命令 Option>Font，可以将字符的尺寸变成“ 小 ”、“ 正常 ”或“ 大 ”。

5.2 对象和对象等级

5.2.1 对象和对象等级

与 Windows Explorer 的文件夹和文件的结构相同，SIMATIC 管理器可以在 STEP 7 的项目和库中显示对象等级。

下图是一个对象等级的例子。



- 项目对象
- 站对象
- 编程模块对象
- S7/M7 程序对象
- 源文件文件夹对象
- 块文件夹对象

对象有下列功能：

- 对象属性的载体
- 文件夹
- 功能的载体（如：启动某特殊应用）

对象作为属性的载体

对象同时携带功能和属性（如设定）。当选择某一对象，则能够执行下列功能：

- 用菜单命令Edit>Open Object，编辑对象
- 用菜单命令Edit>Object Properties，打开对话框，设定对象的特定选择。

文件夹也可以做为属性的载体。

对象做为一个文件夹的形式

文件夹（子目录）可能含有其它的文件夹或对象。当你把它打开后，则可以显示这些内容。

对象作为功能的载体

当你打开一个对象以后，则显示一个可编辑的窗口。

- 一个对象，或是一个文件夹，或是一个功能的载体。除非是这样一个站：它即是文件夹（编程模式），也是功能载体（用于硬件组态）。
- 当你双击某个站的图标，其中的对象就显示出来：编程模式和站的组态（站作为一个文件夹）。

当你用菜单命令 Edit>Open Object 打开一个站时，你就能对其进行组态和参数设定（站作为一个功能载体）。当你双击“ Hardware ”对象时，与菜单命令有着相同的效果。

5.2.2 项目对象

项目位于对象等级最高层，它包括一个自动化解决方案中的所有的数据及程序。

在项目视窗中的位置



- 项目对象
- 站对象
- 编程模块对象
- S7/M7 程序对象
- 源文件文件夹对象
- 块文件夹对象

图　标	对象文件夹	重要功能选择
	项目	• 建立项目 • 项目和库的归档 • 打印项目文献 • 再排列 • 翻译并编辑操作文本 • 插入操作站对象 • 多个用户编辑项目 • 转换版 1 项目 • 转换版 2 项目 • 设置 PG/PC 接口

图标	在项目级的对象	重要对象
	站 SIMATIC 300 SIMATIC 400	• 插入一个站点 • 这个图标即表示一个站也表示一个文件夹（站的等级），其它功能可以在站对象下找到
	S7 程序 M7 程序	• 插入 S7/M7 程序 • S7/M7 程序图标即是对象（项目级），也是对象文件夹（程序级）。其它功能可以在 S7/M7 程序对象中找到
	网络 用于网络组态和设定属性的启动工具	• 通讯站点和分支网络的属性 • 总览：整体数据通讯 • 组态整体数据通讯过程

5.2.3 库对象

库由 S7/M7 程序组成，用于存贮软件块。库位于对象等级的上层。



- 库对象
- S7/M7 程序对象
- 源文件文件夹对象
- 块文件夹对象

图标	对象文件夹	重要功能选择
	库	• 标准库总览 • 有关程序库 • 项目和库的归档

图标	库级中的对象	重要功能选择
	S7 程序 M7 程序	• 插入 S7/M7 程序 • S7/M7 程序图标即是对象（项目级），也是对象文件夹（程序级）。其它功能可以在 S7/M7 程序对象中找到

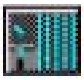
5.2.4 站对象

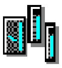
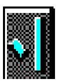
SIMATIC 300/400 站用一个或多个程序模式表示 S7 的硬件组件。

在项目视窗中的位置



- 项目对象
- 站对象
- 编程模块对象
- S7/M7 程序对象
- 源文件文件夹对象
- 块文件夹对象

图标	对象文件夹	重要功能选择
	站	• 插入一个站点 • 上载站参数 • 下载组态参数到可编程控制器 • 从一个站点上载组态参数 • 显示 CPU 报文和用户定义的诊断报文 • 组态 “ 系统错误报告 ” • 诊断硬件，显示模板信息 • 显示和改变操作模式 • 显示并设置时间和日期 • 清除装载存储器和工作存储器，初始化 CPU
	SIMATIC PC 站	• 生成并设定 SIMATIC PC 站点参数 • 组态 SIMATIC PC 站间的通讯结构

图标	站点级中的对象	重要功能选择
	硬件	• 硬件组态基本步骤 • 站点组态基本步骤 • 总览：组态设定本机配置的参数 • 组态 DP 主站系统基本步骤 • 组态多计算操作
	编程模块	• 编程模块既是对象（站点级）又是对象文件夹（“ Programmable Modules ” 级）。其它功能可以在编程模块对象中找到

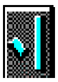
5.2.5 编程模块对象

编程模块可以再现设定的参数（CPUxxx，FMxxx，CPxxx。无记忆的系统数据（例如 CP441 通过站点 CPU 装入。因此，没有“系统数据”的对象在项目等级中不能显示。

在项目视窗中的位置



- 项目对象
- 站对象
- 编程模块对象
- S7/M7 程序对象
- 源文件文件夹对象
- 块文件夹对象

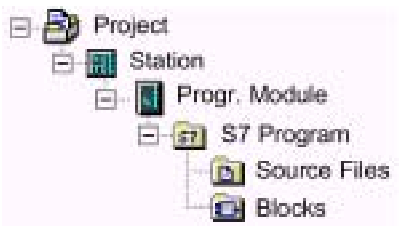
图标	站点级中的对象	重要功能选择
	编程模块	• 总览：组态设定本机配置的参数 • 显示 CPU 报文和用户定义的诊断报文 • 组态“系统错误报告” • 诊断硬件，显示模板信息 • 通过 EPROM 存储卡下载 • 设定访问可编程控制器的口令 • 显示强制数值窗口 • 显示和改变操作模式 • 显示并设置时间和日期 • 设定操作属性 • 清除装载存贮区和工作暂存区，初始化 CPU • 在线视窗中的诊断符号 • 内存分区 • 将下载软件块存入集成 EPROM • 升级可编辑控制器中的操作系统

图标	编程模块级对象	重要功能选择
	程序： S7 程序 M7 程序 程序	• 插入 S7/M7 程序 • S7/M7 程序图标即是对象(Project 级)，也是对象文件夹（Program 级）。其它功能可以在 S7/M7 程序对象中找到。
	设定网络	• 项目中的网上站点 • 联接类型及站点 • 明确联接的不同类型 • 输入新的联接 • 组态用于 SIMATIC 站中的模板联接

5.2.6 S7/M7 程序对象

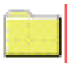
S7/M7 程序是一个文件夹，它包括 S7/M7 CPU 软件或非 CPU 的软件（如：CP 或 FM 模板）。

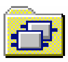
在项目视窗中的位置



- 项目对象
- 站对象
- 编程模块对象
- S7/M7 程序对象
- 源文件文件夹对象
- 块文件夹对象

图标	对象文件夹	重要功能选择
	S7 程序	• 插入 S7-/M7-程序 • 建立地址优先权 • 建立逻辑软件块 • 设定信息号码 • 生成并编辑用户定义的诊断报文 • 翻译并编辑操作文本 • 显示 CPU 报文和用户定义的诊断报文 • 程序测试

图标	对象文件夹	重要功能选择
	M7 程序	• M7 系统程序
	程序	• 在项目中生成软件（一般）

图标	程序级中的对象	重要功能选择
	对单一或多个变量设定符号的符号表	• 绝对地址和符号地址 • 符号表的结构及元素 • 输入共享符号 • 输入符号时的提示 • 设定和编辑与符号相关的信息 • 翻译并编辑用户文本 • 用符号表组成控制与监测 • 编辑通讯属性 • 符号表输入/输出接口
	源文件	• 其它功能可以在源文件夹对象中找到
	软件块文件夹	• 其它功能在软件块文件夹对象中寻找

5.2.7 块文件夹对象

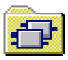
离线方式软件块文件夹包括：逻辑软件块（OB，FB，FC，SFB，SFC），数据块（DB），用户定义数据类型（UDT）和变量。系统数据对象表示系统数据块。

在线方式的软件块文件夹包括可下载到可编程控制器并可执行的程序。

在项目视窗中的位置



- 项目对象
- 站对象
- 编程模块对象
- S7/M7 程序对象
- 源文件文件夹对象
- 块文件夹对象

图标	对象文件夹	重要功能选择
	软件块	• 用 Project Management（项目管理）下载 • 不用 Project Management（项目管理）下载 • 可用参考数据概述 • 再接线 • 软件块的比较 • 翻译并编辑操作文本 • 转换软件块的系统属性、语言描述及提示

符号	块文件夹对象	重要功能选择
	软件块概要	• 建立逻辑软件块 • 生成软件块 • 编写 STL 源文件的基本信息 • 软件块的比较
	组织块（OB）	附加功能： • 引入数据及参数类型 • 下载条件 • 用程序状态功能进行测试 • 你应了解的关于在单步模式/断点下进行测试的信息 • 再接线 • 软件块的注释

符号	块文件夹对象	重要功能选择
	功能（FC）	附加功能： • 引入数据及参数类型 • 下载条件 • 用程序状态功能进行测试 • 你应了解的关于在单步模式/断点下进行测试的信息 • 再接线 • 软件块及参数属性
	功能块（FB）	附加功能： • 引入数据及参数类型 • 使用多重背景 • 下载条件 • 用程序状态功能进行测试 • 你应了解的关于在单步模式/断点下进行测试的信息 • 再接线 • 软件块及参数属性 • 设定和编辑与软件块相关的信息 • PCS 7 报文组态 • 翻译并编辑用户文本 • 设定 FB 参数的系统属性
	用户定义的数据类型（UDT）	• 生成软件块 • 编写 STL 源文件的基本信息 • 引入数据及参数类型 • 使用用户定义的数据类型访问数据 • 软件块及参数属性
	数据块（DB）	• 数据块中的数据总览 • 数据块的声明表显示状态 • 下载条件 • 编写数据块状态 • 引入数据及参数类型 • 使用多重背景 • 软件块及参数属性 • 设定和编辑软件块的相关信息（只限于背景数据块） • PCS7 报文组态（仅限于背景数据块） • 翻译和编辑用户文本（仅限于背景数据块）
	系统功能(SFC)	• 下载条件 • 软件块及参数属性 • 软件块的注释

符号	块文件夹对象	重要功能选择
	系统功能块（SFB）	• 下载条件 • 软件块及参数属性 • 设定和编辑与软件块相关的信息 • PCS 7 报文组态 • 翻译并编辑用户文本 • 软件块的注释
	变量表（VAT）	• 用变量表进行监视和修改的基本程序 • 测试变量表 • 监测变量 • 修正变量 • 强制变量
	系统数据块（SDB）	系统数据块（SDB）仅能用下列功能编辑： • 硬件组态 • 通讯站点和分支网络的属性 • 总览：整体数据通讯 • 设定和编辑与符号相关的信息 • 下载条件

5.2.8 源文件文件夹对象

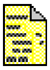
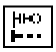
源文件文件夹含有文本方式的源程序。

在项目视窗中的位置



- 项目对象
- 站对象
- 编程模块对象
- S7/M7 程序对象
- 源文件文件夹对象
- 块文件夹对象

图标	对象文件夹	重要功能选择
	源文件夹	• 编写 STL 源文件的基本信息 • 导出源文件 • 导入源文件

图标	源文件夹对象	重要功能选择
	源文件 (如：STL 源文件)	• 编写 STL 源文件的基本信息 • 生成 STL 源文件 • 将软件块模式插入 STL 源文件 • 将源代码插入 STL 源文件 • 检查 STL 源文件的一致性 • 编译 STL 源文件 • 从软件块生成 STL 源文件 • 导出源文件 • 导入源文件
	网络模板	• 生成网络模板 • 在程序中插入一个段模板

5.2.9 没有站点或 CPU 的 S7/M7 编程

在没有组态 SIMATIC 站之前，你可以编写程序。这也就意味着初始工作与你要编程的站点无关。

生成S7/M7程序

1. 用菜单命令 File > Open 或项目窗口，打开相应的项目。
2. 选择离线方式项目窗口中的项目。
3. 根据在可编程控制器上生成的程序，选择下列菜单命令：
   Insert > Program > S7 Program，如果你的程序要运行在SIMATIC S7设备上。
   Insert > Program > M7 Program，如果你的程序要运行在SIMATIC M7设备上。

这时，S7/M7 程序已进入 “ Project ” 窗口。它包括一个程序文件夹和一个空的符号表，在此你可以建立，编写软件块。

编写程序

你可以编写与某种程序模式无关的程序，而后再用移动或复制方式，将程序移到相应模式中。

添加程序到库中

如果你需要多次使用某个 SIMATIC S7 的程序，并将其设置成软件工具，可将这个程序插入库中。但是测试时，一定将其放入项目中，因为这是程序与可编程控制器联接的唯一途径。

访问可编程控制器

选择项目的在线方式。在含有程序属性的对话框中设定地址。

提示

当删除站点或程序模式的时候，你会得到是否删除其中程序的提示。如果你选择不删除程序，则这个程序以无站点的形式直接插入项目。

5.3 用户接口与操作

5.3.1 操作原理

目的：简化面向对象的操作

图形方式的用户接口，使软件的操作更加直观。你可以用日常生活中熟悉的方式在软件中找到所需对象，例如站点、模板、程序、软件块。

包括对象的建立、选择及操作，都可以在 STEP 7 软件平台上完成。

面向应用的操作

以应用为导向的操作，你必须明确哪类操作用于哪类任务，然后启动。

而以对象为导向的操作仅须确定使用哪种对象。而后打开这个对象，对其进行编辑。

使用对象为导向的操作，无须知道指令的具体含意对象以图标的形式作为用户操作界面。用户只须用菜单命令或光标点击则可以打开对象。

当你打开一个对象以后，其相应的应用软件或编辑内容会自动显示出来。

请阅读...

下面几页讲述编辑对象的基本步骤。因为在手册中，对此没有详细说明，请用心详读下列部分。

5.3.2 窗口内容排列

标准窗口内容排列如下：



标题栏和菜单栏

标题栏和菜单栏位于窗口的最上端。标题栏包括窗口以及标题和控制窗口的图标。菜单栏包括窗口所有有效菜单。

工具栏

工具栏由图标（或工具按钮）组成，这些图标以快捷键方式作为经常使用的菜单命令用鼠标点击执行。当用光标选中某个快捷键时，在状态栏会有简单的信息显示。

如果某些键不能操作，则呈灰色。

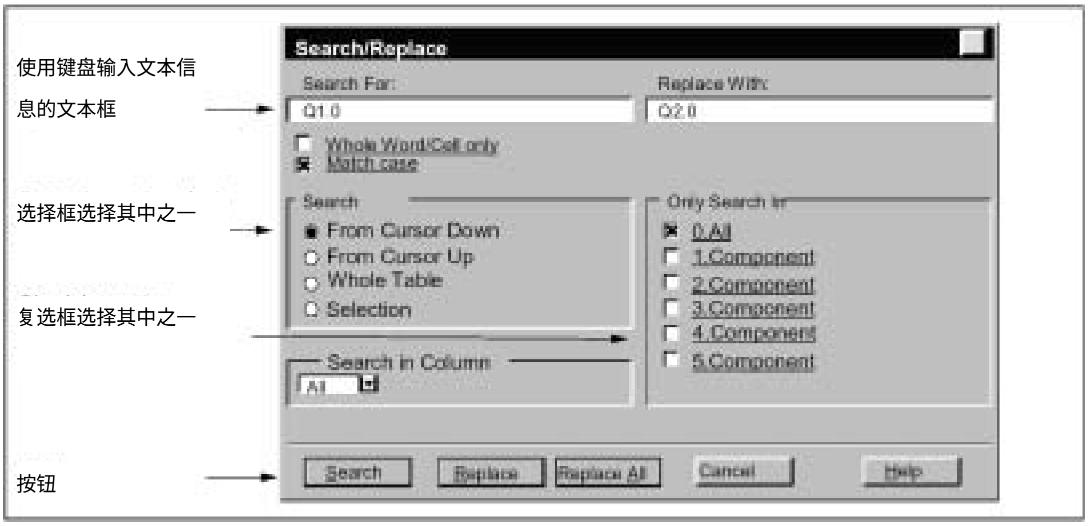
状态栏

状态栏显示相关信息。

5.3.3　对话框中的元素

在对话框中输入信息

你可以将某些特定任务信息输入对话框。经常显示的部分如下图所示：

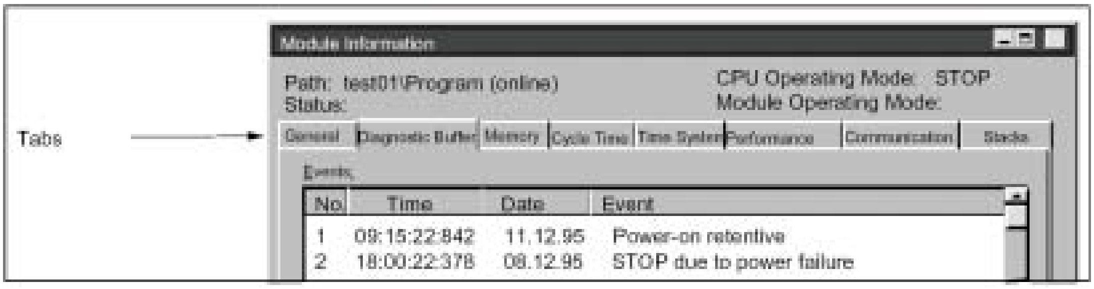


列表框和组合框

有时文本框旁边带有一个向下的箭头。这个箭头表示还有更多的选择。点击这个箭头，则显示一个列表框或组合框。如果用光标点击其中之一，则它可以自动显示在文本框中。

对话框中的选项卡

某些对话框由选项卡组成，通过将对话框分为选项卡，来改进信息的透明度（如下图）。



选项卡出现在对话框的最上端。你只要用光标点击某个选项卡，这个选项卡就会凸出来。

5.3.4 对象的建立和管理

无论哪一种对象基本操作步骤都是相同的，基本操作顺序在手册中另有说明，在此汇总如下：

- 建立对象
- 选择对象
- 在对象执行操作（如复制，删除）

设定建立新的Projects/Libraries的路径

在第一次建立一个新的 Project/Libraries 之前，用菜单命令 Options>Customize，建立你所希望的对象存在的路径。用对话框的“General”选项卡，建立 Project/Libraries 的路径名。

建立对象

利用 STEP 7 Wizard “ New Project ”，建立新的项目，并且插入对象。使用菜单命令 File > “ New Project ” Wizard，打开 Wizard，对话框中将显示你所建立的项目结构，而后利用 Wizard，建立项目。

如果不用 Wizard，菜单命令 File>New，也能建立项目和库。这些对象形成对象体系的起始部分。你可以用插入菜单建立那些不能自动建立的对象。此外，用硬件组态或“ New Project ” Wizard，在 SIMATIC 站建立模板对象。

打开对象

打开对象有多种方式，如下所示：

- 双击对象图标
- 选择对象及菜单命令Edit > Open Object。这只适用于文件夹中没有的对象。

一旦打开对象，你就能生成或改变它的内容。

当打开一个不含有其它对象的一个对象时，这个对象的内容是由可编辑的软件形成一个新的窗口。你不能改变这个对象，因为其中内容已被调用。

提示

例外：站点以文件夹的方式显示编程模块（当你双击它们）和站点组态。当你双击“ Hardware ” 对象，硬件组态功能开始启动。选择这个站点和选择菜单命令 Edit>Open Object 效果相同。

建立对象体系

用“ New Project ” Wizard 建立对象体系。打开一个文件夹，它所包括的对象也就显示出来。这时你可以用插入菜单建立更多的对象，如：在 Project 中增加站点。只有将对象插入到当前文件夹的命令，才能在插入菜单中调用。

设定对象属性

对象属性是指对象本身的一些数据，这些数据决定对象的性能特点。当建立一个新的对象，设定对象属性的对话框会自动显示，用于对象属性的设定。对象属性也可以后期改变。

调用菜单命令 Edit>Object Properties，打开对话框或设定对象属性。

调用菜单命令 Edit>Special Object Properties，打开对话框，输入有关操作控制，监测功能及组态信息。

例如：为了显示一个块的特殊对象属性。其属性关于操作控制及监测功能，这个块必须置有相应的操作控制及监测功能的符号，也就是将系统属性“ S7_m_c ”文件通过块属性的“ Attribute ”键设为“ true ”。

提示

“ System Data ”文件夹和“ Hardware ”对象的属性不能显示或改变。

不能在只读 Project 对话框中输入数据。这时输入框为灰色。

当显示编程模块时，由于一致性的原因，你这时不能编辑这些显示的参数。为此，应调用“ Configuring Hardware ”功能。

剪切、粘贴、复制

所有的对象都可以在窗口下剪切、粘贴、复制。也可以在编辑菜单中找到这类指令。

你也可以用光标拖、放的方法，复制对象。如果错误地将其移动或复制到一个非法的目标，光标就会显示一个禁止的报警信号。

当复制某个对象的时候，这个对象体系下面的所有内容都将被复制。这样你为某个自控工程所建立在该对象的内容就可以多次使用。

提示

“ Connection ”文件夹中的相关表格不能被复制。注意这时你所复制的操作相关文本表，目标对象仅能接受它自己具有的语言形式。

在复制对象时，将一步一步地提示你进行复制。

对象更名

SIMATIC 管理器为每一个新的对象设计有一个标准名。这个名字是由对象的类型（如果在相同的文件夹中也可以生成同样的对象代码）和代码组成。

例如：第一个 S7 程序名为“ S7 Program（1）”，第二个名为“ S7 Program（2）”等。符号表名为 “ Symbols ”，因为它在每一个文件夹中仅存在一个。

你也可以改变对象和 Project 的名字，使其新名字与内容相关。

同 Project 一样，路径名不能超过 8 个字符。否则，当存档和使用 “ C for M7 ”（Borland 编译程序）时，就会出现问题。

你可以直接地或用对象本身属性改变对象的名字。

- 用直接方式：
  用鼠标点击对象名两次，这时对象名边框就会显示出来。你再用键盘输入一个新的名字。
- 使用对象属性：
  选择相应对象及菜单命令Edit>Object Properties。在对话框中改变它的名字。当关闭对话框后，这个对象的名字被修改，并在其下方显示出来。

如果某个对象名字不能修改，则该对话框的输入区域呈灰色，显示当前名字，不能输入文本信息。

提示

当编辑名字或执行其它操作（例如：选择菜单命令）时，如果你将光标移到对象名框之外，这个操作就会结束。但修改的新名字还是可以被接受。

在 “ Rename Objects ” 菜单下，会有详细的提示

移动对象

你可以利用 SIMATIC 管理器，将对象从一个文件夹移到另一个不同 Project 的文件夹。当移动一个文件夹的时候，它的所有内容都同时被移走。

提示

下列对象不能移动：

- 联接的
- 在线显示的系统数据块（SDB）
- 在线显示的系统功能（SFC）和系统功能块（SFB）

在移动对象时，在“ Moving Objects ”下会有详细提示。

对象分类

依据对象属性对其分类预览（菜单命令 View>Details）。为此，用鼠标点击相关的属性标题。当再次点击时，对象的顺序就会反过来，软件块由它们自身的数码顺序分类，例如：FB1、FB2、FB11、FB12、FB21、FC1。

对象预置分类

当再次打开 Project 时，将根据预置分类次序显示对象。如：

- 软件块显示依次为：“ 系统数据、OB、FB、FC、DB、UDT、VAT、SFB、SFC。”
- 在Project里面，首先显示所有的站，而后是S7 程序。

因此，预置分类不能以字母增减的顺序详细预览。

预置分类顺序的再分类

例如：再分类后，点击标题“ Object Name ”上端，就能对预置分类进行再分类，具体操作顺序如下：

- 在详细预览时点击“ Type ”。
- 关闭Project，而后再打开。

删除对象

你可以删除一个文件夹及对象。当删除一个文件夹的时候，其中的所有对象也同时被删除。

你不能取消删除操作。当你不能确认某个对象是否保留，最好首先将整个 Project 存档。

提示

下列对象不能删除：

- 联接的
- 在线显示的系统数据块（SDB）
- 在线显示的系统功能（SFC）和系统功能块（SFB）

在删除对象时，在“ Deleting Object ”下会有详细提示。

5.3.5 在对话框中选择对象

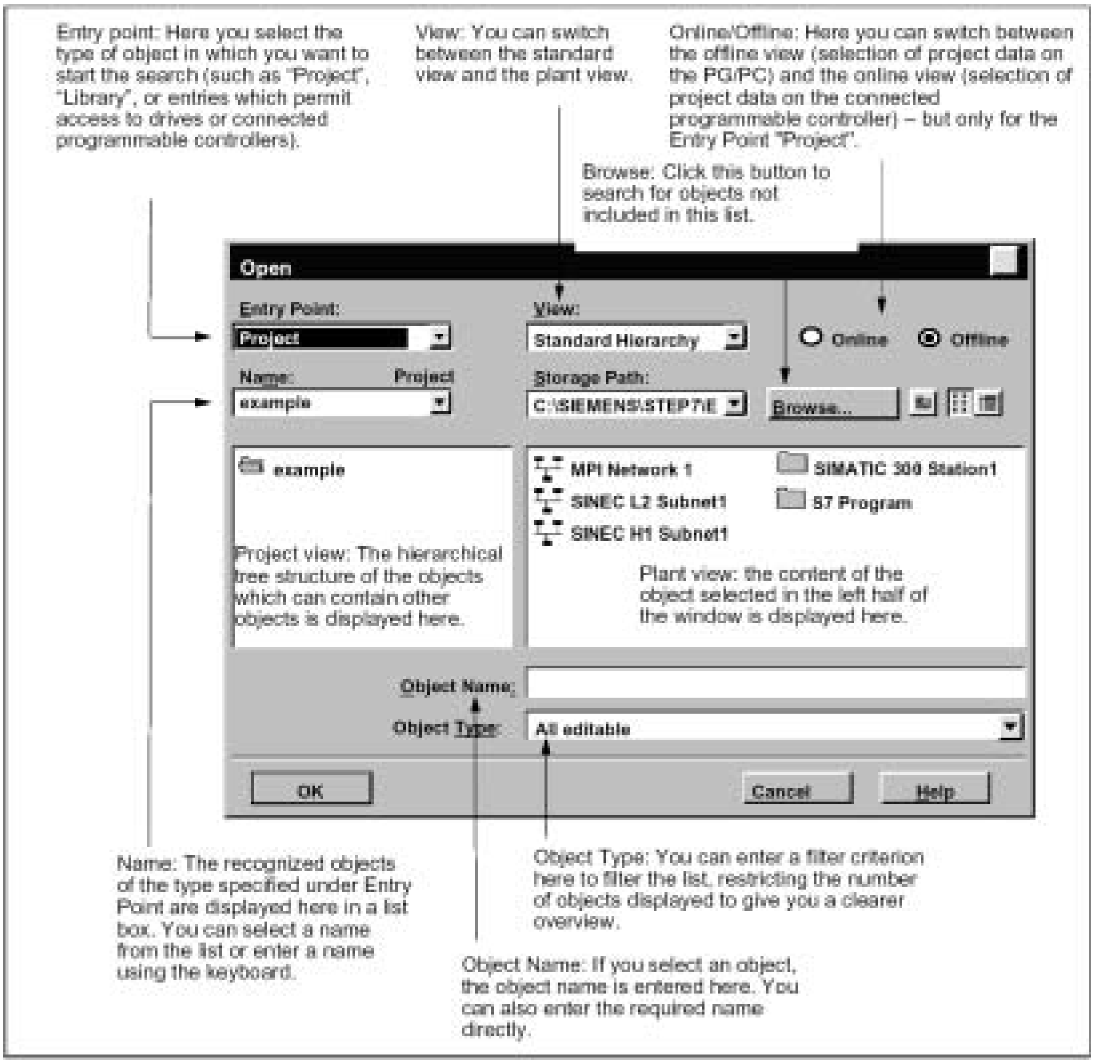
用浏览器对话框选择对象，一般需要进行多步骤的操作。

调用浏览器

你可以用硬件组态操作方式调用浏览器对话框，如：用菜单命令 Station>New /Open（“ SIMATIC Manager ”窗口除外）。

浏览器对话框的结构

在对话框中有下列部分可供你选择。



Entry point: Here you select the type of object in which you want to start the search (such as "Project", "Library", or entries which permit access to drives or connected programmable controllers).
View: You can switch between the standard view and the plant view.
Online/Offline: Here you can switch between the offline view (selection of project data on the PG/PC) and the online view (selection of project data on the connected programmable controller) – but only for the Entry Point "Project".
Browse: Click this button to search for objects not included in this list.
Open
Entry Point:
Project
View:
Standard Hierarchy
Online
Offline
Name:
Project
example
Storage Path:
C:\SIEMENS\STEP7\E
Browse...
example
MPI Network 1
SINEC L2 Subnet1
SINEC H1 Subnet1
SIMATIC 300 Station1
S7 Program
Project view: The hierarchical tree structure of the objects which can contain other objects is displayed here.
Plant view: the content of the object selected in the left half of the window is displayed here.
Object Name:
Object Type:
All editable
OK
Cancel
Help
Name: The recognized objects of the type specified under Entry Point are displayed here in a list box. You can select a name from the list or enter a name using the keyboard.
Object Type: You can enter a filter criterion here to filter the list, restricting the number of objects displayed to give you a clearer overview.
Object Name: If you select an object, the object name is entered here. You can also enter the required name directly.

5.3.6 时间段存贮器

SIMATIC 管理器可以存贮窗口的内容（打开的 Project 和 Libraries）及排列方式。

- 你可以用菜单命令Options > Customize，决定在时间段结束时是否存贮窗口内容及排列。下一个时间段开始时恢复这些内容。在打开的Project画面上，光标位于最后选择的文件夹。
- 用菜单命令Window > Save Settings，存贮实时运行的窗口内容及排列。
- 你可以用Window > Restore Settings命令，恢复用Window > Save Settings存贮的窗口内容及排列方式。在打开的“Project”画面上，光标位于最后选择的文件夹。

提示

在线“Project”的窗口内容、“Accessible Nodes”的窗口内容和“S7 Memory Card”的窗口内容不能保存。访问可编程控制器（S7-300/S7-400）的口令，在时间段结束时不能存贮。

5.3.7 改变符号表的窗口排列

如果你想排列所有包含打开的符号表的窗口，让它们叠放并能看见每个窗口的标题，选择菜单命令 Window > Arrange > Cascade。

如果你想从上到下排列所有包含打开的符号表的窗口，选择菜单命令 Window ＞Arrange＞Horizontally。

如果你想从左到右排列所有包含打开的符号表的窗口，选择菜单命令 Window ＞Arrange＞Vertically。

如果你想沿主窗口的下边沿排列最小窗口的图标，选择菜单命令 Window＞Arrange Icon。

5.3.8 窗口排列的存贮及恢复

STEP 7 具有这样的特性：存贮实时窗口的排列。在下一次显示时按存贮的排列方式显示窗口。可以使用菜单命令 Options > Customize“General”选项卡完成。

存贮什么内容?

当存贮窗口排列时，存贮下列信息：

- 主窗口的位置，
- 打开的Project和Libraries及它们的窗口位置
- 窗口级联的次序

提示

在线“Project”窗口，“Accessible Nodes”窗口和“S7 Memory Card”窗口不能存贮。

存贮窗口排列

用菜单命令 Window > Save Settings，保存当前窗口排列。

恢复窗口排列

用菜单命令 Window>Restore Settings，恢复窗口排列

提示

当恢复窗口时，只能显示已存贮的对象部分等级的内容。

5.4 键盘控制

5.4.1 键盘控制

国际键名	德国键名
HOME	POS 1
END	ENDE
PAGE UP	BILD AUF
PAGE DOWN	BILD AB
CTRL	STRG
ENTER	Eingabetaste
DEL	ENTF
INSERT	EINFG

5.4.2 用于菜单命令的组合键

用 ALT 键和组其它键组合，可任意选择菜单命令。

依次按下例键：

- ALT键
- 菜单中带下划线的字符（如：ALT，F用于“ File ”），如果菜单栏中含“ File ”的话。这个菜单就会打开。
- 菜单命令中带有下划线的字符（如ALT N用于“ New ”菜单命令）。如果这个菜单命令中含有子菜单，子菜单也同时打开。如上述方式可以选择所有菜单命令。

一旦输入组合健的最后一个字符，菜单命令就开始执行。

如：

菜单命令	组合键
File > Archive	ALT，F，A
Window > Arrange > Cascade	ALT，W，A，C

菜单命令快捷键

命令	快捷键
New （File Menu）	CTRL+N
Open （File Menu）	CTRL+O
Close （File Menu）	
Compile （File Menu）	CTRL+B
Print （Object） （File Menu）	CTRL+P
Exit （File Menu）	ALT+F4
Copy （Edit Menu）	CTRL+C
Cut （Edit Menu）	CTRL+X
Paste （Edit Menu）	CTRL+V
Delete （Edit Menu）	DEL
Select All （Edit Menu）	CTRL+A
Object Properties （Edit Menu）	ALT+RETURN
Open Object （Edit Menu）	CTRL+ALT+O
Download （PLC Menu）	CTRL+L
Operating Mode （PLC Menu）	CTRL+I
Update （View Menu）	F5
Updates the status display of the visible CPUs in the online view	CTRL+F5
Customize （Options Menu）	CTRL+ALT+E
Reference Data, Display （Options Menu）	CTRL+ALT+R
Arrange, Cascade （Window Menu）	SHIFT+F5
Arrange, Horizontally （Window Menu）	SHIFT+F2
Arrange, Vertically （Window Menu）	SHIFT+F3
Context-Sensitive Help （Help Menu）	F1 （如果选择菜单命令，则显示其提示内容。否则只显示提示起始画面）

5.4.3 光标移动组合键

在菜单栏/弹出菜单中移动光标

功能	按键
移到菜单栏	F10
移到弹出菜单	SHIFT+F10
移到带有下划线字符或数字的菜单	ALT+菜单中带有下划线的字符
选择带有下划线字符的命令	菜命令中带有下划线的字符
移向左边菜单命令	左箭头
移向右边菜单命令	右箭头
移向上边菜单命令	上箭头
移向下边菜单命令	下箭头
执行菜单命令	ENTER 键
放弃菜单名或关闭打开的菜单并返回	ESC 键

编辑文本时光标的移动

功能	按键
向上移一行或如文本仅有一行向左移一个字符	上箭头
向下移一行或如文本仅有一行向右移一个字符	下箭头
左移一个字符	左箭头
右移一个字符	右箭头
左移一个字	左箭头
右移一个字	右箭头
移到行首	HOME
移到行尾	END
翻转到上一页	PAGE UP
翻转到下一页	PAGE DOWN
移到文本起始位位置	CTRL+HOME
移到文本起终止位置	CTRL+END

在对话框中移动光标

功能	按键
从一个对话框到另一个对话框（从左到右，从上到下）	TAB
在对话框内以反方向移动	SHIFT+TAB
选择带下划线字符菜单	ALT+菜单中带有下划线的字符
选择选项列表	箭头键
打开选项列表	ALT+下箭头键
选择或放弃菜项标题	空格键
确认输入并关闭对话框（“ OK ”键）	ENTER
关闭对话框对修正不做存贮（“ Cancel ”按钮）	ESC

5.4.4 选择文本的组合键

选择或放弃文本	按键
右移一个字符	SHIFT+右箭头键
左移一个字符	SHIFT+左箭头键
移到行的起始位置	SHIFT+HOME
移到行的终止位置	SHIFT+END
移到上一行	SHIFT+上箭头键
移到下一行	SHIFT+下箭头键
翻转到上一页	SHIFT+PAGE UP
翻转到下一页	SHIFT+PAGE DOWN
移到文本文件起始位位置	CTRL+SHIFT+HOME
移到文本文件起终止位置	CTRL+SHIFT+END

5.4.5 访问在线帮助的组合键

功能	按键
打开帮助功能	F1（如正在执行菜单命令，则显示相应帮助否则显示功能初始画面）
选择带有问号的帮助键	SHIFT+F1
关闭帮助窗口并返回	SHIFT+F4

5.4.6 用复合键完成窗口切换

功能	按键
选择不同的窗口	F6
如没有相应窗口，返回前一级窗口	SHIFT+F6
注释窗口与保留窗口（变量表窗口间的切换）。如果没有保留窗口，则返回前一级窗口	SHIFT+F6
注释窗口间的切换	CTRL+F6
返回到前一级注释窗口	SHIFT+CTRL+F6
在注释窗口间的切换（应用系统和保留窗口），当返回这一系统后则激活最后打开的注释窗口	ATL+F6
返回前一级注释窗口	SHIFT+ALT+F6
关闭时实运行窗口	CTRL+F4

6　创建并编辑项目

6.1　项目结构

项目可用来存储为自动化任务解决方案而生成的数据和程序。这些数据被收集在一个项目下，包括：

- 硬件结构的组态数据及模板参数
- 网络通讯的组态数据
- 为可编程模板编制的程序

生成一个项目的主要任务就是为编程准备这些数据。

数据在一个项目中以对象的形式存储。这些对象在一个项目下按树状结构分布（项目层次）。在项目窗口中各层次的显示与 Windows 95 资源管理器中的相似。只是对象图标不同。

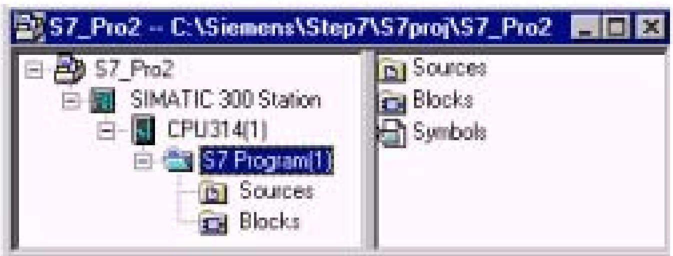
项目层次的顶端结构如下：

1 层：项目
2 层：子网、站或 S7/M7 程序。
3 层：依据第二层中的对象而定。

项目窗口

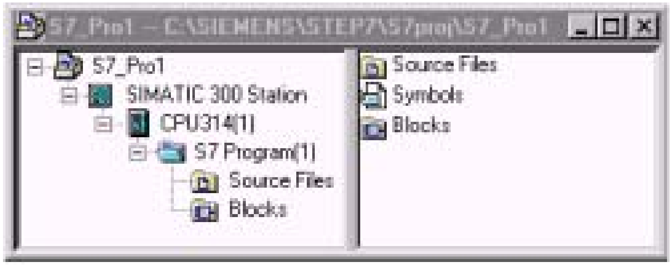
项目窗口分成二个部分。左半部显示项目的树状结构。右半部窗口以选中的显示方式（大符号，小符号，列表），或明细数据显示左半窗口中打开的对象中所包含的各个对象。

在左半窗口点击“ + ”符号以显示项目的完整的树状结构。最后的结构看起来就象下图一样。



在对象层次的顶层是对象“ S7_Pro1 ”作为整个项目的图标。它可以用来显示项目特性并以文件夹的形式服务于网络（组态网络），站（组态硬件），以及 S7 或 M7 程序（生成软件）。当选中项目图标时，项目中的对象显示在项目窗口的右半部分。位于对象层次（库以及项目）顶部的对象在对话框中形成一个起始点用以选择对象。

项目查看



在项目窗口中，你可以通过选择“ offline（离线） ”，显示编程设备中该项目结构下已有的数据，也可以通过选择“ online（在线） ”，通过该项目显示可编程控制系统中已有的数据。

如果安装了相应的可选软件包，你还可以设置另外一种查看方式：设备查看。

注意

硬件及网络的组态只能在“ offline ”下进行。

6.2 建立一个项目

6.2.1 建立项目

使用项目管理结构来构造一个自动化任务解决方案，你需要生成一个新的项目。新项目应生成在你的“ General ”菜单中为项目设定的路径下，该操作可通过菜单命令 Options>Customize 选中。

注意

SIMATIC 管理器允许名字多于八个字符。但是，由于在项目目录中名字被截短为八个字符，所以一个项目名字的前八个字符应区别于其它的项目。名字不必区分大小写。

无论是手动生成项目还是使用向导（Wizard）生成项目，你都会找到每一步骤的向导。

使用向导生成一个项目

生成一个新项目最简单的办法是使用“ New Project（新项目）”向导。使用菜单命令 File>“ New Project ” Wizard，打开 Wizard，对话框中将显示你所建立的项目结构，向导会注意你在对话框中输入所要求的详细内容，然后生成项目。除了硬件站、CPU、程序文件、源文件夹、块文件夹及 OB1，你甚至还可以选择已有的 OB 作故障和过程报警的处理。

下图所示为使用向导生成的项目



手动生成项目

你还可以在 SIMATIC 管理器中，使用菜单命令 File>New，生成一个新的项目。它已包括“ MPI Subnet（MPI 网络） ”对象。

可选步骤

当你编辑项目时，大部分任务的执行顺序是可以灵活掌握的。一旦生成了一个项目，接下来你可以选择以下的任一方法：

- 首先组态硬件，然后为它生成软件程序，或
- 先生成一个与组态的硬件无关的软件程序。

可选方法1：先组态硬件

如果你想先组态硬件，可参照《STEP 7 手册》第二卷中组态硬件部分进行。如果已经组态硬件，组态硬件完成后，生成软件所需的“ S7 Program ”和“ M7 Program ”文件夹则已插入。接下来，继续插入编程所需的对象。

可选方法2：先生成软件

你可以在没有硬件组态的情况下先生成软件；然后再组态硬件。对于程序编辑来说，并不需要将站的硬件结构事先设好。

基本步骤如下：

1. 在项目中插入所需的无站或CPU S7软件文件夹S7/M7程序。
2. 在这儿你可以决定是否在程序文件夹中包含S7硬件或M7硬件。
3. 然后就可以为可编程模板生成软件了。
4. 组态硬件。
5. 一旦完成硬件组态，就可以将M7或S7程序与CPU联系起来。

6.2.2 插入一个站点

在项目中，站代表着可编程控制器的硬件结构，它包含着每一个模板的组态数据及参数赋值。

用“New Project（新建项目）”向导生成的新项目中已经包含了一个站。另外，你可以用菜单命令 Insert>Station，生成站。

你可在下列各种站中作选择：

- SIMATIC 300站
- SIMATIC 400站
- SIMATIC H 站
- SIMATIC PC 站
- PC/Programming device（可编程设备）
- SIMATIC S5
- 其它站，即，非SIMATIC S7/M7及SIMATIC S5

站在插入时带有预置名（如，SIMATIC 300 Station（1），SIMATIC 300 Station（2））等。如果愿意的话，你可以用一个相应的站名替代预置名。

在帮助“Inserting a Station（插入一个站）”下面，你可以找到一步步插入一个站的向导。

组态硬件

当你组态硬件时，可以借助于模板样本对可编程控制器中的 CPU 及各模板进行定义。你可以通过双击站来启动硬件组态的应用程序。

一旦你存储并退出硬件组态，对于在组态中生成的每一个可编程模板，都会自动生成 S7 或 M7 程序及连接表（“Connections”对象）。用“New Project”向导生成的项目则已包含这些对象。

在帮助“Configuring the Hardware（组态硬件）”下面，你能够找到一步一步组态的向导，更详细的信息见帮助“Basic Steps for Configuring a Station（组态站的基本步骤）”。

生成连接表

每一个可编程模板可自动生成一个（空）的连接表（“ Connections ” 对象）。连接表可用来定义网络中可编程模板之间的通讯连接。打开连接表，则有一个表格窗口显示出来，你可以在这里定义可编程模板之间的连接。

在帮助“ Networking Stations within a Project（在一个项目中连网各站）”下面，你可以得到更详细的信息。

下一步骤

一旦完成了硬件组态，你可以为可编程模板生成软件（见帮助“ Inserting a S7/M7 Program（插入 S7/M7 程序）”）。

6.2.3 插入 S7/M7 程序

为可编程模板编制的软件存储在对象文件夹中。对 SIMATIC S7 模板而言，该对象文件夹称作“ S7 Program ”，对 SIMATIC M7 模板，它则被称作“ M7 Program ”。

下图是在一个 SIMATIC 300 站中可编程模板的 S7 程序的示例。



现已存在的组件

每个可编程模板都会自动生成一个 S7/M7 程序来存储软件；

在新生成的 S7 程序中，以下对象已经存在：

- Symbol table符号表（“ Symbols ” 对象）
- “ Blocks（块）”文件夹，用于存储第一个块
- “ Source Files（源文件）”文件夹，用于生成源文件

在新生成的 M7 程序中，以下对象已经存在：

- Symbol table符号表（“ Symbols ” 对象）
- “ Blocks ” 文件夹

生成S7块

要用语句表、梯形图、或功能块图生成程序。可选择已经存在的“ Blocks ” 对象，然后选择菜单命令 Insert>S7 Block。在子菜单中，你可以选择想要生成的块的类型（如：数据块、用户定义的数据类型（UDT）、功能、功能块、组织块或变量表）。

你可以打开一个（空）的块，然后用语句表、梯形图或功能图输入程序。你可以在语句表、梯形图及功能图手册里，在生成逻辑块的基本操作的章节中得到更多的相关信息。

注意

在用户程序中，可能会有由系统生成的“ 系统数据 ”（SDB）。你可以打开它，但为了保持一致性，你不能修改它。一旦你装载，并将修改后的内容下载到可编程控制器就会改变组态。

使用标准库中的块

你可以使用软件提供的标准库中的块来生成用户程序。使用菜单命令 File>Open，可以访问库。你可以在“ Working with Libraries（使用库进行工作）” 以及在线帮助中得到更多的有关使用库及生成自己的库的信息。

生成源文件/CFC图表

如果你想用某种特定的编程语言生成一个源文件或 CFC 图表，可选择 S7 程序中的对象“ Source Files ” 或“ Charts ”，然后选择菜单命令 Insert>S7 Software。在子菜单中选择与你的编程语言相配的源文件。现在可以打开一个（空）的源文件输入程序了。你可以在“ Basic Information on Programming in STL Source Files（STL 源文件的基本编程信息）” 中获得更多的信息。

生成M7程序

如果你想为 M7 系列的可编程模板的 RMOS 操作系统编制程序，可选择 M7 程序。然后选择菜单命令 Insert>M7 Software。在子菜单中，可选择与你的编程语言或操作系统相匹配的对象。现在，你可以打开所生成的对象并访问相应的编程环境。

生成符号表

当生成一个 S7/M7 程序时会自动生成一个（空）符号表（“Symbols”对象）。打开符号表时，“符号编辑器”窗口将显示一张符号表，可在该表中定义符号。你可以在“Entering Multiple Shared Symbols in the Symbol Table（在符号表中输入多个共享符号）”中得到更多的信息。

插入外部源文件

你可以用任何 ASCII 编辑器生成并编辑源文件。然后将这些文件引入到项目中，并且编译生成各个块。

将引入的源文件进行编译，所生成的块存储在“Blocks”文件夹中。

你将在“Inserting External Source Files（插入外部源文件）”中获得更多的信息。

6.3 编辑项目

6.3.1 编辑项目

打开一个项目

要打开一个已存在的项目，可选择菜单命令 File>Open，在随后的对话框中选中一个项目，然后，该项目窗口就打开了。

注意

如果你想要的项目没有显示在项目列表中，点击“Browse”按钮。在这里可以搜寻包括已列在项目列表中的项目在内的所有项目。可以使用菜单命令 File>Manage 改变选项。

拷贝一个项目

使用菜单命令 File>Save As，可以将一个项目存为另一个名字。

你可以使用菜单命令 Edit>Copy，拷贝项目的部分如：站，程序，块等。

你可以在“ Copying a Project and Copying Part of a Project （拷贝项目及项目的一部分）”中找到拷贝项目操作的向导。

删除一个项目

使用菜单命令 File>Delete，可删除一个项目。

使用菜单命令 Edit>Delete，可删除项目中的一部分，比如：站，程序，块等。

你可以在“ Deleting a Project and Deleting Part of a Project（删除项目及删除项目的一部分）”中找到删除项目的操作步骤。

6.3.2 管理多语言文本

使用 STEP 7，可以导出使用一种语言在项目中生成的文本，并进行翻译，重新导入，并以所翻译的语言显示。

下列文本类型可以进行多语言管理。

- 注释和标题
  - 块标题和块注释
  - 网络标题和网络注释
  - STL 程序行注释
  - 符号表、变量声明表、用户定义数据类型和数据块注释
  - HiGraph 程序注释、状态名和转移名
  - S7-Graph 程序中的步名和步注释扩展
- 显示文本
  - STEP 7、S7-Graph、S7-HiGraph 或 S7- PDIAG 生成的报文文本
  - 系统文本库

导出

在所选对象中，导出所有块和符号表。每个文本类型都将生成一个导出文件。该文件包括一列源语言和一列目标语言。源语言中的文本禁止改变。

导入

在导入过程中，目标语言栏的内容（右列）将装入所选项目中。只有与找到的源语言栏中现有文本相匹配的文本，才能被接受。

转换语言

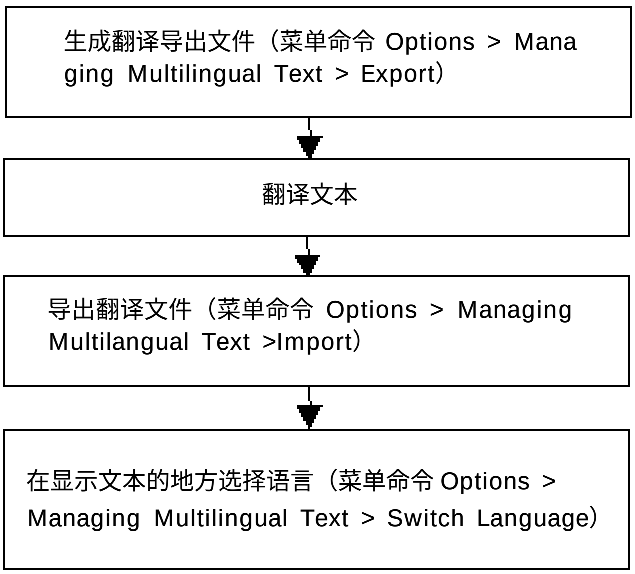
在转换语言时，可以选择导入所选项目时指定的所有语言。然后，改变所选对象语言。

删除语言

在删除一种语言时，该语言中的所有文本都将被从内部数据库中删除。

在项目中应总有一种语言作为基本语言。例如，可以是你的母语。该语言不能删除。在导入和导出时，一定要指定该基本语言为源语言。目标语言可以根据需要设定。

基本程序



7 定义符号

7.1 绝对地址和符号地址

在 STEP 7 程序中，你可以寻址 I/O 信号、存储位、计数器、定位器、数据块和功能块。你可以在程序中用绝对地址来访问这些地些，但是，如果使用符号地址，你的程序读起来会更容易（例如，参照你们工厂的代码系统），使用 Motor_A_On 或其它的标识符。你的用户程序中的地址则可通过这些符号来寻址。

绝对地址

一个绝对地址由一个地址标识符和一个存储地址组成（如，Q4.0，I1.1，M2.0，FB21）。

符号地址

如果你给绝对地址赋予符号名字，则你的用户程序的可读性会更好，故障诊断更容易。

STEP 7 可自动地将符号地址转换成所需的绝对地址。如果要用符号名访问数组、结构、数据块、局域数据、逻辑块以及用户定义的数据类型，必须首先将符号名赋给绝对地址，然后才能对这些数据进行符号寻址。

例如，你可以将符号名 MOTOR_ON 赋给地址 Q4.0，然后在程序指令中就可以使用 MOTOR_ON 寻址了。使用符号地址可以比较容易地辨别出程序中所用操作数与过程控制项目中的元素的对应关系。

注意

在符号名中不允许使用两个连续的下划线（如 MOTOR_ON）。

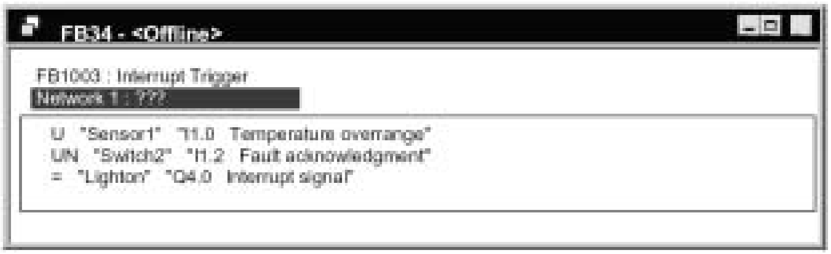
编程支持

在梯形图、功能块图及语句表这三种编程语言的表达方式中，你可以使用绝对地址或符号来输入地址、参数和块名。

使用菜单命令 View>Display>Symbolic Representation，你可以在绝对地址和符号地址两种表达方式之间进行切换。

为使符号地址编程更简单，你可以显示该符号的绝对地址及注释。使用菜单命令 View>Display>Symbol information，可激活此信息功能。这意味着语句表指令后面的行注释包含更多的信息。你无法在显示中进行编辑，任何修改都需在符号表或声明表中进行。

下图所示为语句表中的符号信息。



当打印一个块时，当前屏幕上带有语句注释或符号注释的表达方式会被打印出来。

7.2 共享和局域符号

符号寻址允许你用有一定含义的符号地址来替代绝对地址。将短的符号和长的注释结合起来使用，可以使编程更简单，程序文件作得更好。

你必须区分局域（块定义）符号和共享符号。

	共享符号	局域符号
有效性	• 在整个用户程序中有效， • 可以被所有的块使用， • 在所有的块中含义是一样的， • 在整个用户程序中是唯一的。	• 只在定义的块中有效 • 相同的符号可在不同的块中用于不同的目的
允许使用的字符	• 字母、数字及特殊字符， • 除 0x00，0xFF 及引号以外的强调号 • 如使用特殊字符，则符号须写出在引号内	• 字母 • 数字 • 下划线（_）
使用	你可以为以下各项定义共享符号： • I/O 信号（I，IB，IW，ID，Q，QB，QW，QD） • I/Q 输入与输出（PI，PQ） • 存储位（M，MB，MW，MD） • 定时器（T）/计数器（C） • 逻辑块（FB，FC，SFB，SFC） • 数据块（DB） • 用户定义数据类型（UDT） • 变量表（VAT）	你可以为以下各项定义局域符号： • 块参数（输入，输出及输入/输出参数）， • 块的静态数据， • 块的临时数据。
在哪里定义	符号表	块的变量声明表

7.3 显示共享或局域符号

如下所示，你可以在程序的指令部分区分开共享符号和局域符号。

- 符号表中定义的符号（共享）显示在引号内。
- 块变量声明表中的符号（局域）显示时前面加上“＃”。

你不必输入引号或“＃”。当你以 LAD、FBD 或 STL 方式输入程序时，语法检查自动增加这些字符。

如果你担心会出现混淆，比如同一个符号在符号表和变量声明表中都使用了，那么，你必须明确地输入标志着共享符号的代码（引号）。因为在此情况下，如果没有相应的代码符号一律解释为块定义（局域变量）。

如果符号地址中包含空格，则也需在此符号地址上加上共享符号的代码。

当使用 STL 编辑源文件时，可使用同样的特殊字符及划线。在自由编辑模式下不自动加上代码字符，但是，如果你希望避免混淆，有必要输入共享符号的代码。

注意

使用菜单命令 View>Display>Symbolic Representation，你可以在所声明的符号地址和绝对地址之间进行切换。

7.4 建立地址优先权

在 S7 程序属性的对话框中可以设置如果符号表中作了改变，当块被打开时是符号还是绝对地址具有优先权。在版本 V5 以下的 STEP 7 中绝对地址总是具有优先权的。对于块调用命令 CALL，绝对地址总是具有优先权的。

举例

在一个已存储过的块中有这样一条指令“ A Symbol_A ”，这里的 Symbol_A 是在符号表中为绝对地址“ I 0.1 ”定义的符号。现在符号表作了修改，当再次打开该块。设置地址优先权则会对该指令产生如下所示的影响：

地址优先权	将符号赋值“ Symbol_A=I0.1 ”变为	块打开时的指令	解释
绝对地址	Symbol_A=I0.2	A I 0.1	在指令中显示绝对地址 I0.1，因为不再有符号赋值给该地址
绝对地址	Symbol_B=I0.1	A Symbol_B	在指令中显示仍然有效的绝对地址 I0.1 的新符号名
符号	Symbol_A=I0.2	A Symbol_A	指令不变，显示一条信息，注意你符号赋值关系已改变
符号	Symbol_B=I0.1	A Symbol_A	该指令被标定为错误（红色），因为 Symbol_A 不再有定义

7.5 共享符号的符号表

7.5.1 共享符号的符号表

共享符号在符号表中定义。

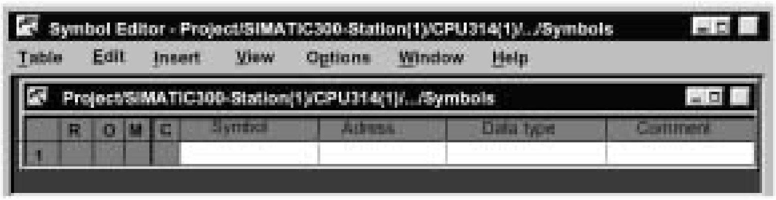
当你生成 S7 或 M7 程序时，一个（空的）符号表（“ Symbols ” 对象）会自动生成。

有效性

符号表只对你的用户程序所连接的模板是有效的。如果你想在几个 CPU 中使用同样的符号，必须确保各符号表中的内容是相配的（比如，拷贝该表）。

7.5.2 符号表的结构及元素

符号表的结构



R/O/M/C 列

R/O/M/C 列显示各符号是否赋予了特殊对象的特性：

- R（监控）意思是指使用任选软件包S7-PDIAG（V5）生成过程诊断错误定义符号。
- O指该符号能够在WinCC下被操作和监控。
- M指该符号被赋予了与符号相关的信息（SCAN）。
- C指该符号被赋予了通讯特性（只能在NCM中被选中）。

Symbol（符号）

符号名不能长于 24 个字符。一张符号表最多可容纳 16380 个符号。

数据块中的地址（DBD，DBW，DBB，DBX）不能在符号表中定义。它们的名字应在数据块声明表中定义。

组织块（OB）、一些系统功能块（SFB）以及系统功能（SFC）已预先被赋予了符号名，当你为 S7 程序编辑符号表时可以引入这些符号名。该引入文件存在 SIMATIC 路径下…\S7data\Symbol\Symbol.sdf。

地址

地址是一个特定存储区域和存储位置的编写，例如，输入 I12.1 例如：Input I 12.1

输入时要对地址的语法进行检查，还要检查看该地址是否可以赋给指定的数据类型。

数据类型

在 SIMATIC 中可以选择多种数据类型。在数据类型区域中已有一个缺省数据类型，如果需要，用户可以修改。如果所作的修改不适合该地址或存在语法错误，当你退出该区域时会显示一条错误信息。

注释

你可以给所有的符号加注释。简短的符号名与更详细的注释混合使用。可使编程更有效，使程序文件更完善。注释最长 80 个字符。

转换为C变量

你可以在 M7 程序的符号表中选中符号，然后将它们转换成相应的与 ProC/C++ 软件选项相连接的 C 变量。

7.5.3 符号表中允许使用的地址和数据类型

在整个符号表中只能使用一套助记符。在 Windows 管理器中使用菜单命令 Options>Customize，在“ Language ”选项中，可以在 SIMATIC（德语）和 IEC（英语）两套助记符之间进行切换。

IEC	SIMATIC	说　明	数据类型	范　围
I	E	输入位	BOOL	0.0 - 65535.7
IB	EB	输入字节	BYTE，CHAR	0- 65535
IW	EW	输入字	WORD，INT，S5TIME，DATE	0 - 65534
ID	ED	输入双字	DWORD，DINT，REAL， TOD，TIME	0 - 65532
Q	A	输出位	BOOL	0.0 - 65535.7
QB	AB	输出字节	BYTE，CHAR	0 - 65535
QW	AW	输出字	WORD，INT，S5TIME，DATE	0 - 65534
QD	AD	输出双字	DWORD，DINT REAL TOD，TIME	0 - 65532
M	M	存储位	BOOL	0.0 - 65535.7
MB	MB	存储字节	BYTE，CHAR	0 - 65535
MW	MW	存储字	WORD，INT，S5TIME，DATE	0 - 65534
MD	MD	存储双字	DWORD，DINT，REAL，TOD，TIME	0 - 65532
PIB	PEB	外设输入字节	BYTE，CHAR	0 - 65535
PQB	PAB	外设输出字节	BYTE，CHAR	0 - 65535
PIW	PEW	外设输入字	WORD，INT，S5TIME，DATE	0 - 65534
PQW	PAW	外设输出字	WORD，INT，S5TIME，DATE	0 - 65534
PID	PED	外设输入双字	DOWRD，DINT，REAL TOD，TIME	0..65532
PQD	PAD	外设输出双字	DOWRD，DINT，REAL，TOD，TIME	0 – 65532
IEC	SIMATIC	说　明	数据类型	范　围
T	T	定时器	TIMER	0 - 65535
C	Z	计数器	COUNTER	0 - 65535
FB	FB	功能块	FB	1 - 65535
OB	OB	组织块	OB	1 - 65535
DB	DB	数据块	DB，FB，SFB，UDT	0 - 65535
FC	FC	功能	FC	0 - 65535
SFB	SFB	系统功能块	SFB	0 - 65535
SFC	SFC	系统功能	SFC	0 - 65535
VAT	VAT	变量表		0 - 65535
UDT	UDT	用户定义数据类型	UDT	0 - 65535

7.5.4 符号表中不完整的和不唯一的符号

不完整的符号

你有可能存储了不完整的符号。比如，先只输入符号的名字，以后再加上相应的地址。这意味着你可能在任何时候中断符号表的编辑，暂时存储这一临时结果，另外再找时间来完成。当你要用符号来编程时（没有错误信息显示），必须事先输好符号名、地址及数据类型。

不唯一的符号是怎样出现的

当你在符号表中插入一个符号，而该符号名和/或地址已在其它行中使用，这时就会出现不唯一的符号。即：新的符号与已存在的符号不唯一。

上述情况会出发生在，比如你为了作一些微小的改变而拷贝并粘贴某一符号。

标定不唯一的符号

在符号表中，以图形加亮（颜色，字符）方式对不唯一的符号进行标定。这种在表达方式上的改变意味着它们仍需要编辑。你可以显示所有符号或有选择地进行显示，即只显示唯一的符号或不唯一的符号。

使符号唯一

当你修改引起符号不唯一的元素（符号和/或地址）时，不唯一的符号就变成了唯一的符号。如果两个符号相互不唯一，当改变其中一个使之成为唯一符号时，另一个也就成了唯一的符号。

7.6 输入共享符号

7.6.1 输入共享符号

有三种方式进行符号输入，这些符号可以在后面的编程当中使用：

- 通过对话框
  在程序输入窗口中你可以打开一个对话框，定义新的符号或对已有的符号重新定义。这种方法最好用于对单个符号的定义，比如，当你编程时发现少了一个符号或想更正某个符号。这种方法你不用显示整张符号表。
- 直接在符号表中
  你可以直接在符号表中输入符号及其绝对地址。这种方法推荐用于编辑大量符号或者为项目生成符号表的情况。因为这种方法会在屏幕上显示已经赋值的符号，你可以更容易地观察这些符号。
- 从其它表格编辑器中引入符号表
  你可以用任何一种你所喜欢的表格编辑器来生成符号数据（比如，Microsoft Excel），然后将该文件引入符号表。

7.6.2 输入符号时的注意

在符号表中输入新符号，将光标点中表中的第一空行并填写该单元。你可以使用菜单命令 Insert>Symbol，在当前行前面插入一个新行。如果光标前面的行已包括有一个地址，在通过“ Address ”和“ Data Type ”栏的预定值插入新的符号时，会受到支持。从前一行中导出地址；缺省数据类型作为数据类型输入。

还可以使用 Edit 菜单中的命令拷贝并修改表中已有各项。存储然后关闭符号表，你还可以保存那些还未完全定义完的符号。

当你在表中输入符号时，须注意以下各点：

列	注意
符号	在整个符号表中名字必须唯一。当你确认该区域的输入或退出该区域时，不唯一的符号则被标定出来。符号名最长可达 24 个字符。引号（"）不允许使用。
地址	当你确认或退出该区域时，程序会自动检查该地址输入是否是允许的。
数据类型	当你确认或退出地址时，该区域被自动地赋予一个缺省数据类型。如果你修改这个缺省类型，程序会检查你的数据类型是否与地址相匹配。
注释	你可以输入注释简单地解释该符号的功能（最多 80 个字符）。输入注释任选。

7.6.3 在对话框中输入单个共享符号

下文所描述的是如何在编辑程序块时利用对话框改变符号或定义新符号而不必显示符号表。

如果你想编辑单个符号，这种方法非常有用。如果你要编辑大量的符号，应打开符号表，直接在里面输入。

在块中激活符号显示

你可以在一个打开的块中用菜单命令 View > Display > Symbolic Reprosentation，可激活符号显示。在菜单命令前有一个对号表示该表达方式激活。

输入程序时定义符号

1. 确认在块的编辑窗口中激活了符号表达方式（菜单命令View > Display > Symboilic Representation）。
2. 在程序指令中选中你想对其进行符号赋值的绝对地址。
3. 选择菜单命令Edit>Symbol。
4. 填写对话框并关闭它，用“OK”确认你的输入并且确保输入的是一个符号。

被定义的符号输入到符号表中。任何导致符号不唯一的输入都将被拒绝并显示错误信息。

在符号表中进行编辑

使用菜单命令 Options>Symbol Table，可打开符号表并进行编辑。

7.6.4 在符号表中输入多个共享符号

打开符号表

有几种方法打开符号表：

- 在项目窗口双击符号表。
- 在项目窗口选中符号表，然后使用菜单命令Edit>Open Object。

处于激活状态的程序的符号表显示在自己的窗口中。此时你可以生成或编辑符号。当你在符号表生成后第一次打开时它是空的。

输入符号

在符号表中输入新符号，将光标点中表中的第一空行并填写该单元。你可以使用菜单命令 Insert>Symbol，在当前行前面插入一个新的空行。还可以使用 Edit 菜单中的命令拷贝并修改表中已有各项。存储然后关闭符号表，你还可以保存那些还未完全定义完的符号。

分类符号

符号表中的数据可以按照符号、地址、数据类型或注释进行按字母的分类。

使用菜单命令 View>Sort，可以打开一个对话框，重新定义分类显示以改变当前表中的分类方式。

筛选符号

你可以用筛选功能在符号表中选择某组数据。

使用菜单命令 View>Filter，可以打开“ Filter ”（筛选）对话框。

你可以定义标准，只有满足标准的数据才能被包括在筛选后的显示当中。可按如下标准进行筛选：

- 符号名、地址、数据类型、注释
- 具有操作员控制及监控属性的符号、具有通讯特性的符号、信息的二进制变量的符号（存储位或过程输入）
- 有效的符号、无效的符号（不唯一、不完整）

各个标准之间通过与操作相连接。筛选后的数据以指定的字符串开始。

如果你想了解“ Filter ”对话框中更多的选项，可以按 F1 打开与之相关的在线帮助。

7.6.5 大小写符号

在大小写字符之间无区别

以前，STEP 7 中定义的符号可能与用于其它情况的符号有所不同。这在 STEP 7 V4.02 中已经进行修改。现在不在有符号之间的差异。

这种修改可以根据客户的需要进行，由此大大减少程序中的错误发生率。符号定义限制也支持 PLCopen forum，以定义转换程序标准。

具有大小写区别的符号定义现在不再支持。在以前，例如，在符号表中可能有下述定义：

Motor1 = I 0.0

motor1 = I 1.0

这些符号的第一个字母都具有大小写区别。这种符号区别很容易造成混淆。新的符号定义避免了这种错误源。

对现有程序的影响

如果你已经使用过不同符号之间的区分标准，你可能对使用新的符号定义有困难，如果：

- 符号只在使用大小写字符时不同。
- 参数只在使用大小写字符时不同。
- 符号与参数只在使用大小写字符时不同。

但是，如下所述，这三种冲突都可进行分析和解决。

符号只在使用大小写字符时不同

冲突：

如果没有使用当前版本软件编辑符号表，在编译源文件时，将使用符号表中不唯一符号的第一个符号。

如果已经编辑符号表，这些编译的符号也无效；意思是指在打开块时不显示符号，并且编译包含这些符号的源文件时，将会出现错误。

排除：

打开符号表，检查符号表是否有冲突，并保存。这操作可以识别非唯一符号。然后，你可以使用“ 非唯一符号 ”滤镜显示非唯一符号，并修正。你还可以修正包

含冲突的任何源文件。由于在打开块时，可以自动使用当前版本（无冲突）符号表，因此无需修改块。

参数只在使用大小写字符时不同

冲突：

在编译包含这些接口的源文件时会出现错误。可以打开使用这些接口的块，但是不能再访问这些参数中的第二个参数。如果你想存取第二个参数，在保存块后，程序将自动返回第一个参数。

排除：

为了检查包含这种冲突的块，建议使用 “ 生成源文件 ” 功能，生成所有程序块的源文件。如果在编译已生成源文件时出现错误，必定有冲突。

应确保参数的唯一性，以校正源文件；例如，通过 “ 查找和替换 ” 功能。然后，再编译文件。

只在使用大小写字符时符号与参数不同

冲突：

如果源文件中的共享符号和局部符号，只在使用在小写字符时不同，并且没有使用首字母大写，以区分共享（“ 符号名 ”）符号或局部（#符号名）符号，在编译时将总是使用局部符号。这会导致修改机器代码。

排除：

在这种情况下，建议生成所有块的新源文件。由此，可自动赋值局域和共享符号的首字母，可保证在将来编译程序时，正确处理。

7.6.6 导入/导出符号表

你可以导出当前的符号表到一个文本文件，这样就可以用任意的文本编辑器进行编辑。

你还可以将其它应用程序中生成的表导入到你的符号表中并继续编辑。这种导入功能可以用于，比如将 S5 程序转换为 S7 之后，将 STEP5/ST 中生成的符号赋值表导入进来。

可选择的文件格式有：*.SDF、*.ASC、*.DIF 以及*.SEQ。

导出规则

可以导出整个符号表、筛选后的部分符号表或表中选中的几行。

用菜单命令 Edit>Special Object Properties，设定的符号特性不能导出。

导入规则

- 为经常使用的系统功能块（SFB）、系统功能（SFC）以及组织块（OB）预先定义的符号表已存在文件……\S7DATA\SYMBOL\SYMBOL.SDF中，需要的话，可以从该文件中导入。
- 当进行导入和导出时，符号的特性不予考虑。符号特性可通过命令菜单Edit>Spectal Object Properties设定。

7.6.7 导入/导出符号表的文件格式

下列文件格式可从符号表中被导入或导出：

- ASCII文件格式（ASC）
- 数据交换格式（DIF）
  可以在Microsoft Excel中打开、编辑并存储DIF文件。
- 系统数据格式（SDF）
  使用SDF文件格式从或向Microsoft Aceess应用程序导入或导出数据。
  - 可以在Microseft Access中打开、编辑并存储SDF文件。
  - 在Access中，选择文件格式“ Text（带分隔符）”。
  - 使用双引号（”）作为文本分隔符。
  - 使用逗号（，）作为单元分隔符。
- 赋值表（SEQ）
  注意：当导出符号表到一个SEQ文件时，注释长于40个字符则第40个字符以后的注释将被截去。

ASCII文件格式（ASC）

文件类型	*.ASC
结 构	数据长度，逗号分隔符，数据
示 例	126，green_phase_ped.　T　2　TIMER Duration of green phase for pedestrians 126，red_Ped.　Q　0.0　BOOL Red for pedestrians

数据交换格式（DIF）

文件类型	*.DIF
结构	一个 DIF 文件包含文件首部及数据

（Header）首部	表（TABLE）	DIF 文件的开始
	0，1	
	“<Title>”标题	注释串
	VECTORS	文件中记录的数量
	0，<No. of records>（记录数）	
	“”	
	TUPLES	记录中的数据区域的数量
	0，<No. of cloumns>（列数）	
	“”	
	DATA	作为首部结束及数据开始的标识
	0，0	
	“”	
数据（每个记录）	<type>（类型）<numeric value>（数字值）	数据类型、数字值的样识
	<string>（字符串）	字母部分或
	V	如果没有使用字母部分

首部： 文件首部必须按规定的顺序包含 TABLE、VECTORS、TUPLES 及 DATA。在 DATA 前面，DIF 文件可以包含更多的可选记录类型。然而这些都被符号编辑器忽略不计。

数据： 在数据部分，每项都包含三个部分：类型的标识（数据类型）、一个数字值以及一个字母部分。

可以在 Microsoft Excel 中打开、编辑并存储 DIF 文件。不必使用重音符、变音符或其它的特殊语言字符。

系统数据格式（SDF）

文件类型	*.SDF
结构：	引号中的字符串，用逗句隔开各部分
示例：	“green_phase_ped”，“T 2”，“TIMER”，“Duration of green phase for pedestrains”， “red_ped”， “Q 0.0”，“BOOL”，“Red for pedestrains”

要在 Microsoft Access 中打开一个 SDF 文件，你应选择 “ Text（带分隔符） ” 文件格式。用双引号（ ”）作为文本的分隔符，用逗号（，）作为区域的分隔符。

赋值表（SEQ）

文件类型	*.SEQ
结构：	TAB 地址 TAB 符号 TAB 注释 CR
示例：	T2 green_phase_ped. Duration of green phase for pedestrains Q0.0 Red_ped. Red for Pedestrains

TAB 代表制表跳格键（09H）

CR 代表 RETURN 键（0DH）回车。

8 程序块和程序库的生成

8.1 选择一种编程方法

根据生成程序时所选用的编程语言，我们可以用增量输入方式输入程序或用自由编辑（文本）方式输入。

增量编辑器适用于梯形逻辑、功能块图、语句表以及S7-GRAPH等编程语言

增量输入方式编辑器适用于梯形图、FBD、STL 和 S7-GRAPH，用该方式生成的块存放在用户程序中。如果你想立刻检查所输入的内容，就应该用增量输入方式。这种编程方式尤其适合于初学者。用增量输入方式输入时，每一行或每个元素都会立即进行句法检查。只有改正了所指出的错误才能完成输入。句法修正过的所有输入经过自动编译存到用户程序中。

任何用于语句中的符号必须事先定义。如果在程序块中使用了没有定义的符号，则该块不能完全编译，但是这种不一致的临时版程序可以存盘。

自由编辑（文本）编辑器适用于语句表、S7 SCL、S7 HiGraph编程语言

在自由编辑方式的编辑器中，是用源文件的形式生成用户程序，该文件再编译成各类程序块。

如果想快速输入程序就应选用自由编辑方式。

在自由编辑方式中，程序或程序块是在文本文件中编辑，然后再对该文本文件进行编译。

文本文件（源文件）存在你的项目中 S7 Program 下的 “ source file ” 文件夹中，例如：STL source file 或 SCL source file。一个源文件可包含一个块或多个块的程序代码。用文本编辑器进行 STL 和 SCL 编程可生成 OB、FB、FC、DB 及 UDT（用户定义数据类型）的代码，也可以生成整个用户程序。CPU 的所有程序（意味着所有的块）可包含在一个文本文件中。

当你编译源文件时，就会生成各种类型的块，并在用户程序中存起来。在文件中使用的任何符号必须在编译之前加以定义。在编译的过程中，由编译器报告错误。

为了顺利地通过编译，在编程语言中使用专门的句法是非常重要的。只有当选择了相容性检查命令或将源文件编译成程序块时，才运行句法检查功能。

8.2 选择编程语言

8.2.1 选择编程语言

为编辑器设置编程语言

当用户要生成某程序块或源文件时，应在对象的属性中设置用于生成该块或源文件的编程语言和编辑器类型。该输入确定当该程序块或源文件打开时，启动的是哪种编辑器。

启动编辑器

在 SIMATIC 管理器中，用双击相应的对象（块，源文件，等），或选择菜单命令 Edit>Open Object，或在工具条中选择相应的按钮，来启动相应的语言的编辑器。

在表中列出的编程语言都可用于生成 S7 程序。在标准的 STEP 7 软件包中包括 LAD、FBD、STL。这几种 STEP 7 编程语言表示类型，也可购买做为可选软件包的其它的编程语言。

你可以选择一系列不同的编程方法（梯形逻辑、功能块图、语句表、高级语言、顺序控制或状态图形）。还可以选择是用文本方式编程，还是用图形方式编程。

选择好编程语言，也就确定了可以用哪种输入方式（·）。

编程语言	用户类	应用	增量输入	自由编辑方式	可从 CPU 备份程序块
语句表 STL	愿意用类似于机器码语言编程的用户	程序在运行时间和存贮空间要求上最优	·	·	·
梯形逻辑 LAD	习惯电路图的用户	编写逻辑控制程序	·	—	·
功能块图 FBD	熟悉布尔代数逻辑图的用户	编写逻辑控制程序	·	—	·
SCL（结构控制语言）可选软件包	用高级语言，如 PASCAL 或 C 编程的用户	数据处理任务程序	—	·	—
S7 Graph 可选软件包	有技术背景，没有 PLC 编程经验的用户	以顺序过程的描述很方便	·	—	·
HiGraph 可选软件包	有技术背景，没有 PLC 编程经验的用户	对异步非顺序过程的描述很方便	—	·	—
CFC 可选软件包	有技术背景，没有 PLC 编程经验的用户	适于连续过程的描述	—	—	—

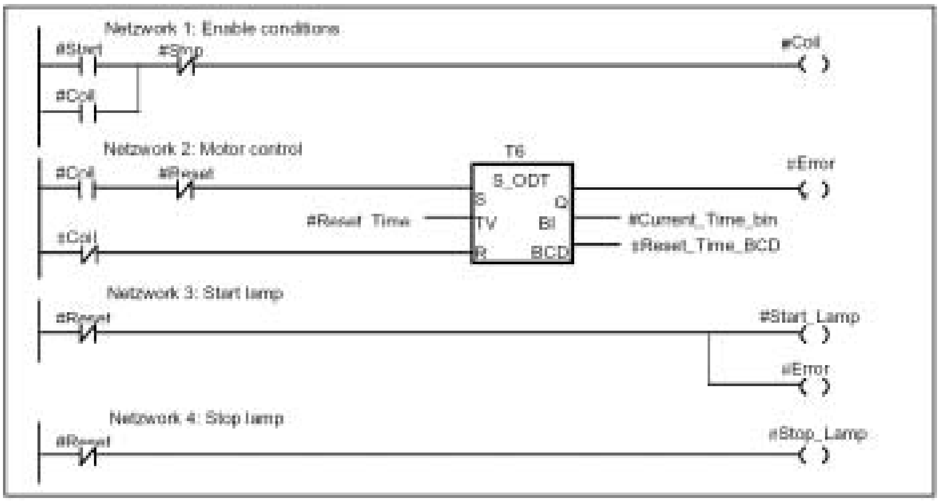
如果程序块中没有错误，可将其在梯形逻辑、功能块图和语句表之间进行切换。如果有部分程序不能切换，则用语句表显示。

可用源文件的语句表生成各程序块，也可将各程序块反编译到源文件中。

8.2.2 梯形逻辑编程语言（LAD）

图形编程语言梯形逻辑是基于电路图表示法的基础之上，在程序段中将电路图中的元素如常开触点和常闭触点组合而成。一个逻辑块的程序部分由一段或多段程序组成。

梯形逻辑程序段举例



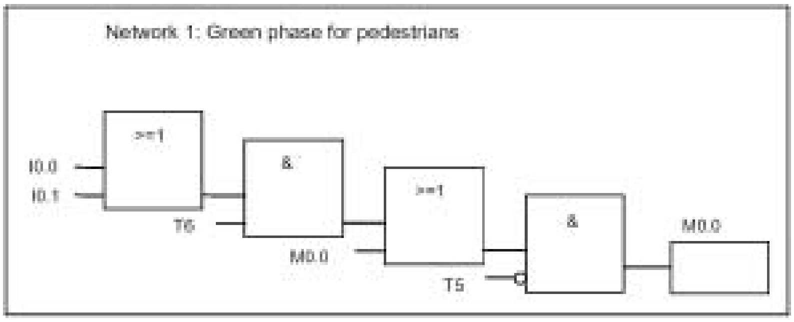
梯形逻辑编程语言包含在 STEP 7 标准软件包中。梯形逻辑程序是用增量编辑器生成。梯形逻辑程序是用增量编辑器生成。

8.2.3 功能块图编程语言（FBD）

编程语言功能块图（FBD）使用类似于布尔代数的图形逻辑符号来表示控制逻辑。一些复杂功能诸如算术功能等，可直接用逻辑框表示。

FBD 编程语言包含在 STEP 7 标准软件包中。

举例：FBD中的一个程序段



在 FBD 方法中用增量编辑器生成程序

8.2.4　语句表编程语言（STL）

编程语言的另一种表示法是语句表，它类似于机器码的一种文本语言。每条语句对应 CPU 处理程序中的一步。多条语句可组成一程序段。

举例：语句表中的程序段

```
        Network 1: Control drain valve
A(
O
O   #Coil
)
AN  #Close
=   #Coil
        Network 2: Display "Valve open"
A  #Coil
=  #Disp_open
        Network 3: Display "Valve closed"
AN  #Coil
=  #Disp_closed
```

语句表编程语言类型包含在 STEP 7 标准软件包中。用这种语言表示类型编辑程序既可以用增量编辑器也可以用 STL 源文件自由模式编辑器，再通过编译将其生成各类程序块。

8.2.5 S7 SCL 编程语言

编程语言 SCL（结构化控制语言）是一个可选软件包，它是按照国际电工技术委员会 IEC1131-3 标准定义的文本语言。ASCAL 类型语言在编写诸如回路和条件分枝时，用其高级语言指令要比 STL 容易。因此，SCL 适合于公式计算，复杂的最优化算法或管理大量的数据。

S7 SCL 程序是用自由编辑方式编辑器中 SCL 源文件生成。

例如：

```
FUNCTION_BLOCK FB20
VAR_INPUT
ENDVAL：                INT；
END_VAR
VAR_IN_OUT
IQ1：                   REAL；
END_VAR
VAR
INDEX：                 INT；
END_VAR

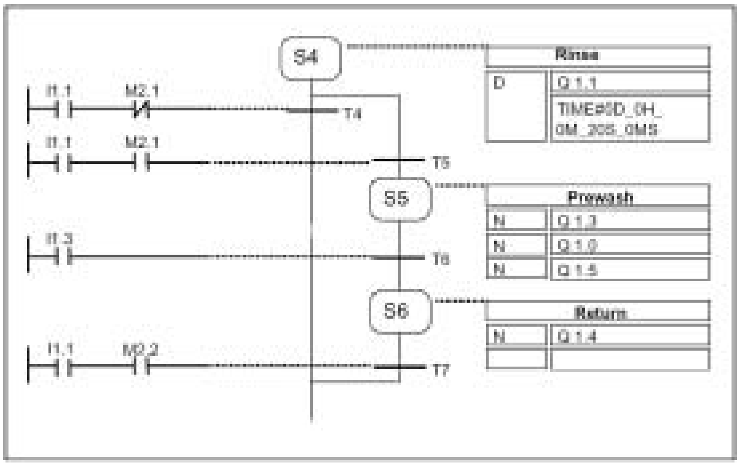

BEGIN
CONTROL：=FALSE；
FOR INDEX：= 1 TO ENDVALUE DO
      IQ1：= IQ1 * 2；
      IF IQ1 >10000 THEN
               CONTROL = TRUE
      END_IF
END_FOR；
END_FUNCTION_BLOCK
```

8.2.6 S7-GRAPH 编程语言（顺序控制）

图形编程语言 S7-GRAPH 属于可选软件包，适用于顺序控制的编程。它包括生成一系列顺序步，确定每一步的内容，以及步与步之间的转换条件。编写每一步的程序要用特殊的编程语言（类似于语句表），转换条件是在梯形逻辑编程器中输入（梯形逻辑语言的流线型版本）。

S7-GRAPH 表达复杂的顺序控制非常清晰，用于编程及故障诊断更为有效。

S7-GRAPH的顺序控制程序举例



程序块的生成

用 S7-GRAPH 编辑器，将生成含有步顺控器的功能块程序。相应的背景数据块中含有顺控器的数据，例如：FB 的参数，顺序步和转换条件。用 S7-GRAPH 编辑器能自动生成背景数据块。

源文件

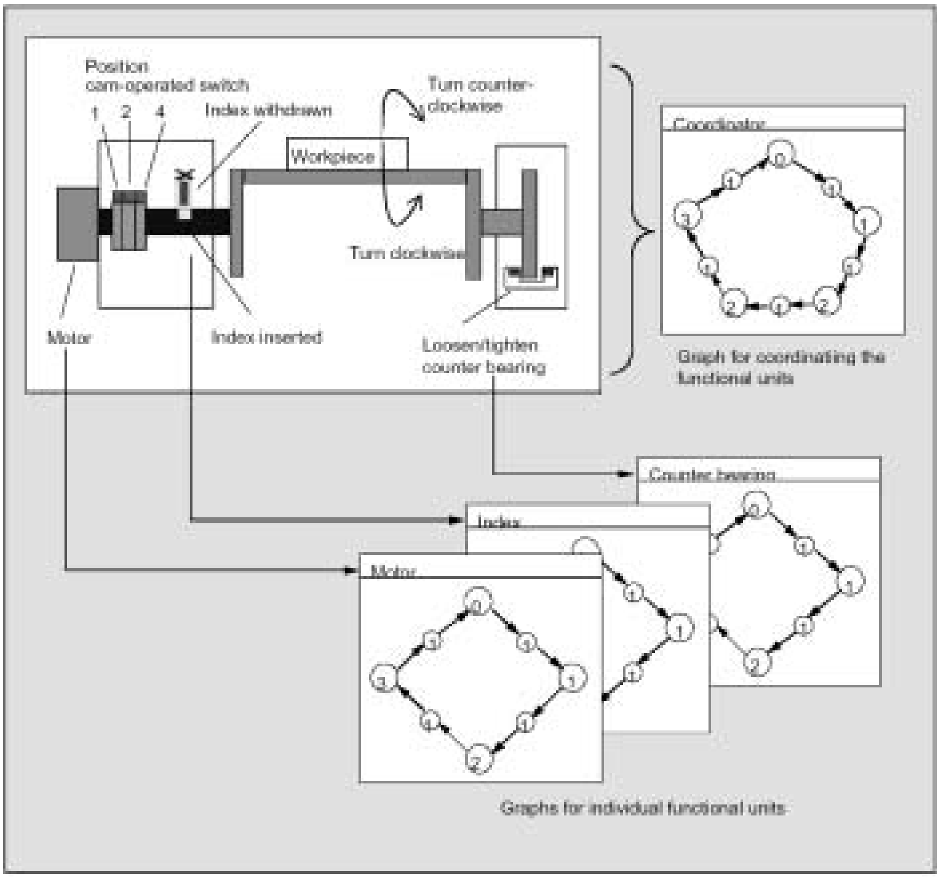
通过 S7-GRAPH 生成的功能块可以产生一个文本源文件（图形源文件），该源文件可由操作员面板（OP）或操作员接口文本显示（TD）编译显示成顺控器。

8.2.7 S7 HiGraph 编程语言（状态图形）

图形编程语言 S7 HiGraph 属于可选软件包，可以将程序中的各块做为状态图形编程。这种方法将你的项目分成不同的功能单元，每个单元有不同的状态。不同状态之间的切换要定义转换条件。用类似于语句表的放大型语言描述赋给状态的功能以及状态之间转换的条件。

每个功能单元都用一个图形来描述该单元的特性。整个项目的各个图形组合起来为图形组。各功能单元的同步信息可在图形之间交换。

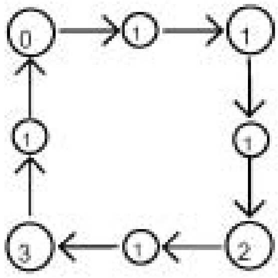
各功能单元的状态条件的清晰表示，使得系统编程成为可能，故障诊断简单易行。与 S7 Graph 不同，在 S7 HiGraph 中任何时候只能一个状态（在 S7 Graph 中：“ 步 ”）是激活的。下列图形为功能单元的图形是怎样生成的（举例）。



状态

从状态 3 至状态 0 的转变

转换条件：



图形组存在 HiGraph 源文件中 S7 program 之下的 “ Source Files ” 文件夹中。该源文件可编译成用户程序中的 S7 程序块。

句法和形式参数在图形最后输入时检查（当工作窗口关闭时）。地址和符号在源文件编译时检查。

8.2.8 S7 CFC 编程语言

可选软件包 CFC（*Continuous Function Chart，连续功能图*），是一种用图形的方法连接复杂功能的编程语言。

编程语言 S7 CFC 用于连接已存在的各种功能。有许多标准功能不需要用户编程，而是可以使用含有标准块（例如：逻辑、算术、控制和数据处理等功能）的程序库。使用 CFC 不需要用户掌握详细的编程知识以及有关可编程序控制方面的专门知识。只需要具有行业所必需的工艺技术方面的知识就可以。

用户生成的程序块可按自己的意愿进行连接，连接的方法分不同的情况，如果用 SIMATIC S7，可用 S7 编程语言中的任一种，如果是用于 SIMATIC M7 则用 C/C++编程语言。

程序是按 CFC 图表生成并存贮。这些程序存在 S7 program 下面的 “ Charts ” 文件夹中。这些图表可编译成用户程序中的 S7 程序块。

8.3 生成软件块

8.3.1 软件块文件夹

用户可以按下列形式生成 S7 CPU 中的程序：

- 软件块
- 源文件

文件夹“ Blocks（软件块） ”在 S7 program 路径之下，用于存放各程序块。

该文件夹中的程序块需要用户下载到 S7 CPU 中用于执行自动控制任务。这些可装载的程序块包括逻辑块（OB，FB，FC）和数据块（DB）块文件夹中会自动生成一个空的组织块 OB1，在 S7 CPU 中必须用该组织块来执行用户程序。

块文件夹还包含下列对象：

- 用户生成的用户定义数据类型（UDT）。UDT可使编程变得容易，但不需要下载到CPU中。
- 用户可生成变量表（VAT）在测试用户程序时用于监视和修改变量。变量表不下载到CPU中。
- 在对象“ 系统数据 ”（系统数据块）中含有系统信息（系统组态，系统参数）。这些系统数据块是当用户进行硬件组态时提供数据并生成的。
- 系统功能（SFC）和系统功能块（SFB），需要时可在用户程序中调用。但是，用户不能自己编写SFC和SFB。

除了系统数据块（只能通过对可编程序控制器的进行组态编辑和生成）以外，用户程序中的其它块都需要用相应的编辑器进行编辑。其编辑器在双击相应的块时自动打开。

注意

用源文件编写的各程序块，当编译后也存在块文件夹之下。

8.3.2 用户定义的数据类型（UDT）

用户定义数据类型是一种特殊的数据结构，由用户自己生成，该结构一旦定义好了之后，可在整个 S7 程序中使用。

- 用户定义数据类型可在逻辑块（FC，FB，OB）变量表中用做数据类型元素或复杂的数据类型。或者在数据块（DB）中做为数据的变量类型使用。其优点是，用户只需定义一次某种常用的特殊的数据结构，然后，可多次使用。
- 用户定义数据类型也可做为样板，用于生成与其数据结构相同的数据块。这就意味着，用户只需生成 一次数据结构，然后，简单地将用户定义数据类型分配给所要生成的数据块。（举例：配方：数据块的结构总是相同的，只是数值不同。）

用户定义数据类型在 SIMATIC 管理器中生成，或在增量编辑器中生成——与其它类型块相同。

用户定义数据类型（UDT）的结构

当你打开一个用户定义数据类型时，出现一个新的工作窗口，在该窗口中用变量表的形式显示用户定义数据类型。

- 在第一行和最后一行已经有STRUCT和END_STRUCT用于表明用户定义数据类型的开始和结束。用户不能编辑这两行。
- 用户在变量表的第二行相应的列中开始输入用户定义数据类型。
- 用户可从下列数据类型中建立用户定义数据类型：
  - 基本数据类型
  - 复杂数据类型
  - 已存在的用户定义数据类型

S7 用户程序中的用户定义数据类型不下载到 S7 CPU 中。既可用增量输入编辑器直接生成和编辑，也可通过源文件编译时生成。

8.3.3 块属性

块属性可使用户更容易辨识所生成的各程序块，并且还可以对这些程序块加以保护，防止非法修改。

用户应在某程序块建立时编辑块属性。除了用户编辑的属性，属性对话框也显示信息：但是用户不能编辑这些信息。

块属性和系统属性也在 SIMATIC 管理器中每块的对象属性中显示。在此用户只能编辑名称，系列，作者和版本等属性。

当通过 SIMATIC 管理器建一个新块时，用户可编辑对象属性。如果不是用 SIMATIC 管理器，而是用其它编辑器生成程序块，则该块的某些属性（如：编程语言）会自动地存入对象属性中。

注意

用于 S7 程序块编程的记忆系统，可通过 SIMATIC 管理器菜单命令 Options > Customize 和 “ Language ” 表设置。

块属性表

当输入块属性时，输入顺序如下表。

关键字/属性	含义	举例
[KNOW_HOW_PROTECT]	块保护；使用该指令后，当该块进行编译时，使程序部分不可见。	KNOW_HOW_PROTECT
[AUTHOR:]	作者名，公司名，部门名或其它名你（最多为没有空格的 8 个字符）	AUTHOR: Siemens, 没有关键字。
[FAMILY:]	块系列名：例如：控制器（最多为没有空格的 8 个字符）	FAMILY: Controllers 没有关键字。
[NAME:]	块名（最多 8 个字符）	NAME: PID 没有关键字。
[VERSION:int1,int2]	块的版本号（两个数的范围均在 0 到 15 之间，意为 0.0 到 15.15）	VERSION: 3.10
[CODE_VERSION1]	功能块能否有多重背景的标识符。如果想用多重背景则功能块不应有该属性	CODE_VERSION1
[UNLINKED]只适用于 DB	某个数据块的属性是 UNLINKED，则不将其连接到程序中	
[READ_ONLY]只适用于 DB	数据块的写保护；其数据只能读，不能修改	FAMILY = Examples VERSIONI = 3.10 READ_ONLY

块保护命令 KNOW_HOW_PROTECT 具有下列效果：

- 如果想在增量STL、FBD或梯形图编辑器中输出编译后的保护块，则该块的程序部分不能显示。
- 该块的参数声明表中只显示类型为 var_in、var_out和var_in_out的变量。类型为var_stat和var_temp的变量隐含。

对应：块属性与块类型

下表所示为哪种块类型可具有哪些块属性：

特 性	OB	FB	FC	DB	UDT
KNOW_HOW_PROTECT	·	·	·	·	—
AUTHOR	·	·	·	·	—
FAMILY	·	·	·	·	—
NAME	·	·	·	·	—
VERSION	·	·	·	·	—
UNLINKED	—	—	—	·	—
READ_ONLY	—	—	—	·	—

当你用源文件编写程序块时，可设置 KNOW_HOW_PROTECT 属性，并在“块属性”对话框中显示，但不能修改。

8.3.4 显示块长度

块长度的显示单位为“bytes（字节）”

块文件夹属性显示

下述长度将显示在离线窗口中的块文件夹属性中。

可编程控制器的装入存储器大小（不包括系统数据的所有块之和）。

可编程控制器的工作存储器大小（不包括系统数据的所有块之和）。

编程器（PG/PC）上的块长度不显示在块文件夹属性中。

块属性显示

在块属性中将显示以下内容：

所需局域数据数量：局域数据按字节表示的大小，

MC7：MC7 按字节表示大小或 DB 用户数据的大小，

编程控制器中输入存储器的大小，

编程控制器中工作存储器的大小只有在识别出硬件分配时，才显示。

显示时，与块是位于在线窗口还是离线窗口中无关。

SIMATIC管理器中显示（详细视图）

如果打开一个块文件夹，并选择了“ Details View（详细视图） ”，工作存储器的要求将显示在项目窗口中，与文件夹是位于在线窗口中还是离线窗口中无关。

选择所有有关块，可以计算块长度的总和。在这种情况下，所选块长度的总和将显示在 SIMATIC Manager 的状态栏中。

对于没有下载到可编程控制器中的块，将不显示其长度（例如变量表）。

编程器（PG/PC）上的块长度不显示在详细视图中。

8.3.5 再接线

下列块和地址可以被再接线：

输入、输出

存储位定时器计数器

功能、功能块

要进行再接线：

1. 在 SIMATIC 管理器中，选择含有要再接线的单个块的“ Blocks ”文件夹。
2. 选择菜单命令 Options＞Rewire。
3. 在“ 再接线 ”对话框中的表中输入所要的替代（旧地址/新地址）。

4. 如果你想要的再接线的地址区域（字节、字、双字），选择选项“ All addresses within the specified address area（指定地址区域内的所有地址）。 ”
例如：你输入IW0和IW4作为地址区域。则地址I0.0-I1.7被再接线为I4.0-I5.7。要进行再接线的区域的地址（如，I0.1）不能够在表中再单独输入。

5. 点击 “ OK ” 按钮。

这就启动了再接线过程。在再接线完成后，你可以在一个对话框中指定是否要看关于再接线的信息文件。该信息文件包含 “ 旧地址 ” 和 “ 新地址 ” 的列表。对每个块还列出了对其所作的接线处理的个数。

在再接线时，应注意以下事项：

再接线块时（即重新命名），不能存在新块。如果存在，过程将中断。

当再接线功能块（FB）时，实例数据块将自动赋值给再接线 FB。实例 DB 不能改变，即，仍保持原有的 DB 号。

8.3.6 软件块及参数属性

有关特性的描述可查找在线帮助中的系统特性。

8.4 有关程序库

8.4.1 有关程序库

程序库用于存放 SIMATIC S7/M7 中可多次使用的程序部件。这些程序部件可从已有的项目中复制到程序库中，也可以直接在程序库中生成。该程序库与其它项目无关。

如果在 S7 program 下的程序库中存放有用户希望多次调用的块，可节省大量的编程时间并提高效率。可以将这些程序块拷贝到用户程序中所需要的地方。

在程序库里生成 S7/M7 程序，与在项目中的做法相同，只是没有测试功能。

生成程序库

生成程序库的方法与生成项目的方法一样，用菜单命令 File>New。新生成的程序库，其目录在菜单命令 Option>Customize 中的“ General ”页设置。

注意

SIMATIC 管理器允许名字多于八个字符。但是，程序库目录名多于 8 个字符将截去。所以各程序库的名称在前 8 个字符中不能相同。名字不必区分大小写。当该目录在浏览器中打开时，可见全名。但当浏览该目录时，则只出现短名。注意，不能在老版本的 STEP 7 项目中，使用新版 STEP 7 程序库中的程序块。

打开程序库

用菜单命令 File>Open 打开一个已存在的程序库。然后选择对话框中的程序库名。最后打开程序库窗口。

注意

如果在程序库表中找不到所需的程序库，则可用“ 打开 ”对话框中的“ Browse ”浏览键。在标准的浏览器窗口中通过显示的目录结构来查找所需的程序库。

注意，文件名总是对应程序库生成时的原始名。在 SIMATIC 管理器中修改文件名不改变文件级的文件名。

当你选择了程序库，则将其列入程序库表中。用户可用菜单命令 File>Manage，修改程序库表中的输入。

复制程序库

用户可用菜单命令 File>Save as，将一个程序库存在另一个名下来复制程序库。

使用菜单命令 Edit>Copy，可复制程序库中的某一部分，如：程序、块、源文件等。

删除程序库

使用菜单命令 File>Delete，可删除一个项目。

用菜单命令 File>Delete，删除程序库。

8.4.2 程序库的等级结构

程序库结构按等级进行，与项目一样：

- 程序库可含有S7/M7 programs。
- S7 program可含有一个“块（Blocks）”文件夹（用户程序）一个“源文件（Source Files）”文件夹，一个“图表（Charts）”文件夹，和一个“符号（Symbols）”对象（符号表）。
- M7 program可含有用于可编程M7模板的图表和C程序以及一个“Symbols（符号）”对象（符号表）和一个“Blocks（块）”文件夹用于数据块和变量表。
- “Blocks（块）”文件夹包含有可下载到各种程序块。S7 CPU.在文件夹中的变量表（VAT）和用户定义数据类型不能下载到CPU中。
- “Source Files（源文件）”文件夹包括各种编程语言生成的源文件程序。
- “Chart（图表）”文件夹包含CFC图表（只有安装了CFC可选软件后才出现）。

当用户插入一个新的 S7/M7 program、“Blocks”文件夹“Source Files”文件夹（只有 S7）和“Symbols”对象会自动插入。

8.4.3 标准库总览

在 STEP 7 标准软件包中包括标准程序库（版本 2/版本 3）：

- stlibs（V2）：标准程序库版本2
- 标准程序标准程序库版本3

标准程序库中含有下列元件：

- 系统功能块：系统功能块（SFB）和系统功能（SFC）
- S5-S7转换块：用于转换STEP 5程序的块
- TI-S7转换块：通用标准功能。
- IEC功能块：用于IEC功能的块，诸如：处理时间和日期信息、比较操作、字符串处理以及选择最大值和最小值。
- 组织块：标准组织块（OB）

标准程序库版本 3 还有下列元件：

- PID控制块：用于PID控制的功能块（FB）
- 通讯块：SIMATICNET CP功能（FC）和功能块。当用户安装可选软件包后，也许还会增加其它的程序库。

删除及安装所提供的程序库

在 SIMATIC 管理器中，用户可以删除所提供的程序库，然后再重新安装。要想安装程序库，用户必须重新执行一次 STEP 7 V5.0 的起动程序。

注意

当用户安装 STEP 7 时，所提供的程序库总是自动安装。如果用户编辑了这些程序库，在重新安装 STEP 7 时，修改过的程序库将会被原始的程序库覆盖。

由于这个原因，用户应在做任何改动之前，复制所提供的程序库，然后只在复制的程序库中进行编辑。

---

9 逻辑块的生成

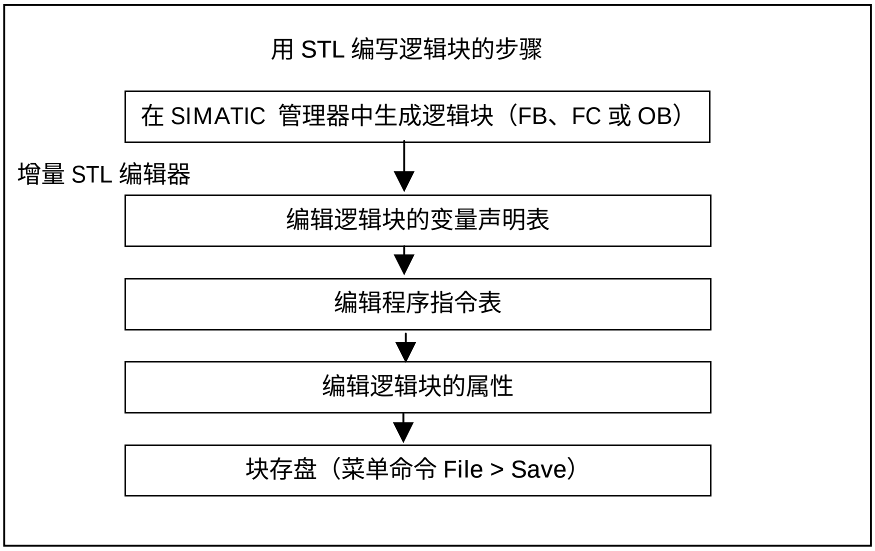
9.1 建立逻辑软件块

9.1.1 建立逻辑软件块

逻辑块（OB、FB、FC）具有变量声明表部分、程序指令部分和属性部分。当用户编程时，必须编辑下列三部分：

- 变量声明表：在变量声明表中，用户可以规定参数及参数的系统特性和本地块特定变量。
- 程序指令部分：在程序指令部分，用户编写能被可编程序控制器执行的块指令 代码。这些程序可分为一段或多段，可用诸如编程语言梯形逻 辑（LAD）、功能块图（FBD）或语句表（STL）来生成程序段。
- 块属性：块属性中有进一步的信息，如：由系统输入的时间标记或路径。此外，用户可输入自己的内容，如：块名、系列名、版本号和作者，并且用户可将系统属性分配给程序块。

原则上说，用户编辑逻辑块各部分的顺序并不重要。当然，各部分可随时修改及增加。



注意

如果用户希望使用符号表中的符号，必须首先检查一下，它们是否存在，需要时可进行修改。

9.1.2 LAD/STL/FBD 程序编辑器的预先设置

在开始编程之前，用户应熟悉一下编辑器的预置表，借助它可使用户在编程时感觉容易并且方便。

用菜单命令 Options>Customize，打开一个分页的对话框。在不同的“ Editor（编辑器）”页中，用户可为逻辑块编程做以下预置，例如“ 逻辑器 ”页中：

- 文本和表格中的字体（类型和大小）。
- 在一个新块中是否显示符号和文字注释。

用户在编辑状态中，用 View 下拉式菜单中的命令可以修改编程语言表示法、文字注释和符号的设置。

在“ LAD/FBD ”页中，用户可以改变主要部分的颜色，例如：程序段或语句行。

9.1.3 逻辑块和源文件的访问授权

当编辑一个项目时，通常使用一个共同的数据库，这就意味着项目组的全体成员也许想在同时访问同一个块或数据源。

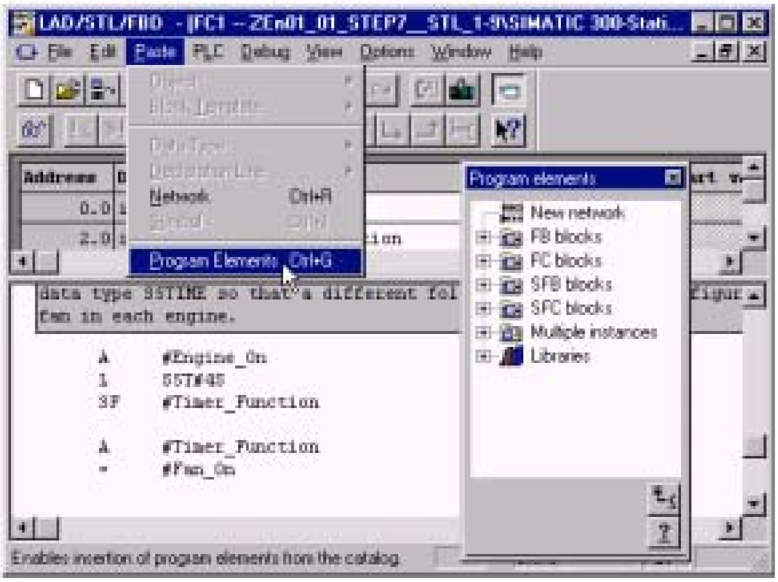
读/写访问授权分配如下：

- 离线编辑：
  当用户试图打开一个块/源文件时，会检查该用户是否对该块有“ 写 ”访问权。如果某块/源文件已经打开，用户只能进行复制。如果用户希望存盘，系统会要求你是否想要覆盖原块，还是存到一个新的名下。
- 在线编辑：
  当用户通过组态连接打开一个在线块时，对应的离线块禁止使用，以保护离线块不会被同时修改。

9.1.4 程序元件目录的指令集

程序元件目录中提供一系列 LAD、STL 和 FBD 中的各元素，以及已经声明的多重背景，已经编好的逻辑块，和程序库中的各逻辑块。目录由菜单命令 View>Catalog 来访问。用菜单命令 Insert>Program Elements，将程序元素插入到程序指令部分中。

STL中程序元素目录举例



9.2 编辑变量声明表

9.2.1 在逻辑块中的变量声明

当用户打开一个逻辑块时，在窗口的上半部分为变量声明表，下半部分为程序指令部分，用户在下半部分编写逻辑块的指令程序。

例如：STL中变量声明表和程序指令部分



在变量声明表中，用户声明在本块中专用的变量包括块的形参和参数的系统属性。这具有以下效果：

- 声明变量后，在本地数据堆栈中为瞬态变量保留一个有效存贮空间，对于功能块，还要为联合使用的背景数据块的静态变量保留空间。
- 当设置输入、输出和输入/输出类型参数时，用户还要在程序中声明块调用的“ 接口 ”。
- 当用户给某功能块声明变量时，这些变量（瞬态变量除外）也在功能块联合使用的每个背景数据块中的数据结构中声明。
- 通过设置系统特性，用户为信息和连接组态操作员接口功能分配特殊的属性，以及参数的过程控制组态。

9.2.2 变量声明表与指令部分之间的关系

逻辑块中的变量声明表和指令部分是紧密联系的，因为在指令部分的程序中要用到变量声明表中的名称。因此，在变量表中的任何变化都将影响整个指令部分的程序。

变量声明表中的改动	指令部分的反应
重新正确输入	使用以前没在变量表中声明的变量，造成无效语句，现在变为有效
变量名改变但类型没变	所有地方的符号立即表示为相应的新名称
正确的变量名改为无效的变量名	指令保持不变
无效的变量名改为正确的变量名	如果有无效语句、变为有效
类型改变	如有无效语句，可变为有效 如有有效语句，可变为无效
删除一个程序中使用的变量（符号名）	有效语句变为无效

文字注释内容的修改，一个新变量的不正确输入、改变初始值或删除一个没用的变量，对指令部分没有影响。

9.2.3 变量声明表的结构

变量声明表中包括：地址、声明类型、符号名、数据类型，初值以及变量的文字注释。表中的每一行表示一个变量声明。数据类型为 ARRAY（数组）或 STRUCT（结构）的变量需要多行。

用户可在逻辑块的本地数据分配数据类型附录表中找到适用于各种块类型的本块数据的有效数据类型。

列	含 义	要 点	编 辑
Address（地址）	按 BYTE.BIT 格式产生地址	对多于一个字节的数据类型，地址用跳到下一个字节地址来表示，关键字为： *：一个数组元素按字节表示的大小 +：与 STRUCT（结构）的开始有关的初始地址 =：STRUCT（结构）要求的完整存贮区	系统输入：地址是由系统进行分配的，并且，当用户结束一个变量声明的输入时显示

列	含　义	要　点	编　辑
Variable （变量）	变量的符号名	变量符号必须以字母开始，不允许用保留的关键字	需要
Declaration (声明类型)	声明类型，变量的“目的”	根据不同的块类型，可以用下列之中的类型 输入参数：“in” 输出参数：“out” In/Out 参数：“in_out” 静态变量：“stat” 瞬态变量：“temp”	根据不同的块类型，由系统提供
Data type (数据类型)	变量的数据类型 (BOOL、INT、WORD,ARRAY 等)	可用鼠标右键弹出菜单再选择合适的元素数据类型	需要
Initial Value (初值)	如果不想用缺省值，可在该栏中设置初值	必须与数据类型匹配。 当数据块第一次存盘时若用户没有明确地声明实际值，则初值将被用于实际值	可选
Comment (文字注释)	用于文档目的的文字注释		可选

预置

当用户打开一个新生成的逻辑块时，显示一个预置的变量声明表。该表只有对应所选块类型按顺序显示的有效参数类型（in、out、in_out、stat、temp）。当用户生成一个新的组织块（OB）时，显示一个标准的变量声明表。该表中的值可改变。

在声明表中不能编辑的列

列	输　入
Address（地址）	地址由系统输入，并当用户结束变量声明时显示
Declaration Type（声明类型）	声明类型是由参数在表中的位置决定的。这就确保变量只能按正确的参数类型的顺序进行输入。如果用户想要改变某一个参数的声明类型，将该参数剪切，粘贴到下面新的声明类型中

9.2.4 有关变量声明表的注意事项

用户可以使用 Edit 菜单中的功能来编辑变量声明表。为使编辑更加容易，请使用鼠标右键的位置敏感菜单。当输入数据类型时，也可用鼠标右键予与支持。

变量声明表中的选择

用户选择某一行时，通过鼠标点击相应的，写保护的地址单元来进行。选择同一声明类型的多行时，用同时按下 SHIFT 键方法。所选中的行颜色变黑。

用户选择 ARRAY（数组）时，通过鼠标点击相应行的地址单元。

如果想选中一个结构（structure），用鼠标选中结构的第一行或最后一行的地址单元（有关键字 STRUCT 或 END_STRUCT 的那一行）。若要选结构中的某一参数，用鼠标点击该行的地址单元即可。

当用户在一个结构中输入另一个结构时，变量名相应地错开，以显示不同的层次。

撤消操作

在变量声明表中，用户可以用菜单命令 Edit>Undo，撤消最近的剪切或删除操作。

9.3 变量声明表中的多重背景

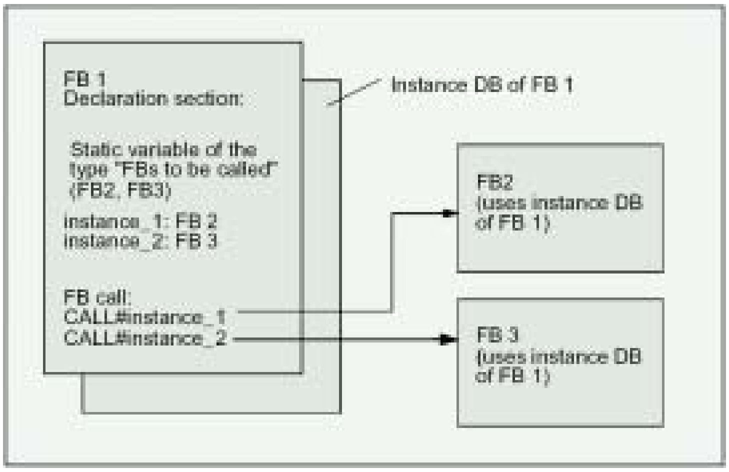
9.3.1 使用多重背景

如果用户希望或不得不用有限的几个数据块存放背景数据以提高 S7 CPU 中的性能（例如：存储能力）是可能的。如果在你的用户程序中调用其它已经存在的功能块（FB）的多层调用，用户可调用这些功能块，而不需要它们自己的（额外的）背景数据块。

解决的方法如下：

- 在调用功能块的变量声明表中，将被调用的功能块做为静态变量参数。
- 在该功能块中，调用其它功能块不带其自己的（额外的）背景数据块。
- 这就将背景数据都压缩在一个背景数据块中，意味着，用户能够更有效地使用数据块。

下面的例子所示为：FB2 和 FB3 使用调用它们的 FB1 的背景数据块。



唯一的要求：用户必须“ 告诉 ”调用功能块，哪个背景你要调用以及这些背景的类型（FB 是什么?）。这些细节必须在调用功能块的参数声明部分输入。被调用的功能块在数据区中至少要有一个变量或参数 VAR_TEMP 不能用。

使用多重背景数据块时，当 CPU 正在运行时不能在线修改。使用背景数据块时，只能保证无冲撞重装。

9.3.2 声明多重背景的规则

多重背景的声明有下列规则：

- 只有在版本2以上的STEP 7中生成的功能块（参看功能块的属性中的块特性），才可能声明多重背景。
- 为了声明多重背景，功能块必须设置为有多重背景能力（在STEP 7中缺省设置），可在编辑器中用Options>Customize取消。
- 必须有一个背景数据块分配给声明了多重背景的功能块。
- 多重背景只能声明为静态变量（声明类型为“ Stat ”）。

注意

- 用户也可以为系统功能块生成多重背景。
- 如果功能块生成时不具备多重背景的能力，而你又想增加这个功能，可以用该功能块产生一个源文件，然后将块属性中的CODE_VERSION1删除，再重新编译该功能块即可。

9.3.3 在变量声明表中输入多重背景

1. 打开功能块，在该功能块中将调用下一级功能块。
2. 如果你不想给被用调的功能块使用背景数据块，可以为这些功能块的每一次调用在调用它们的功能块的变量声明表中定义一个静态变量。
   - 将光标放在“ stat（静态） ”声明的一个空行的第二栏中。
   - 在声明类型“ stat ”的后面，在“ Name（名） ”栏中为FB的调用输入一个名字。
   - 在“ Type（类型） ”栏中输入你想调用的功能块作为一个绝对地址或用它的符号名。
   - 你可以在注释栏输入任何需要的解释。

在程序部分调用

当你已声明了多重背景，你可以调用 FB 和无需指定一个背景 DB。

例如：If the static variable "Name: Motor_1 , Data type: FB20" is defined, the instance can be called as follows:

Call Motor_1　　// Call of FB20 without instance DB

9.4 编辑语句和文字注释时的注意事项

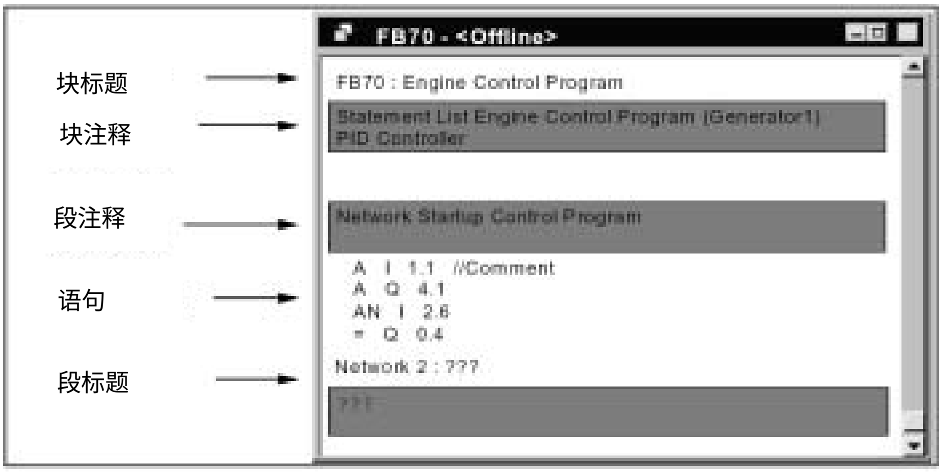
9.4.1 程序指令部分的结构

在程序指令部分，用户通过在程序段中用所选的编程语言输入相应的语句按顺序完成逻辑块中的程序。当语句输入进去时，编辑器立即起动句法检查，发现的错误用红色和斜体显示。

逻辑块的程序指令部分通常由若干段组成，而这些段又由一系列语句组成。

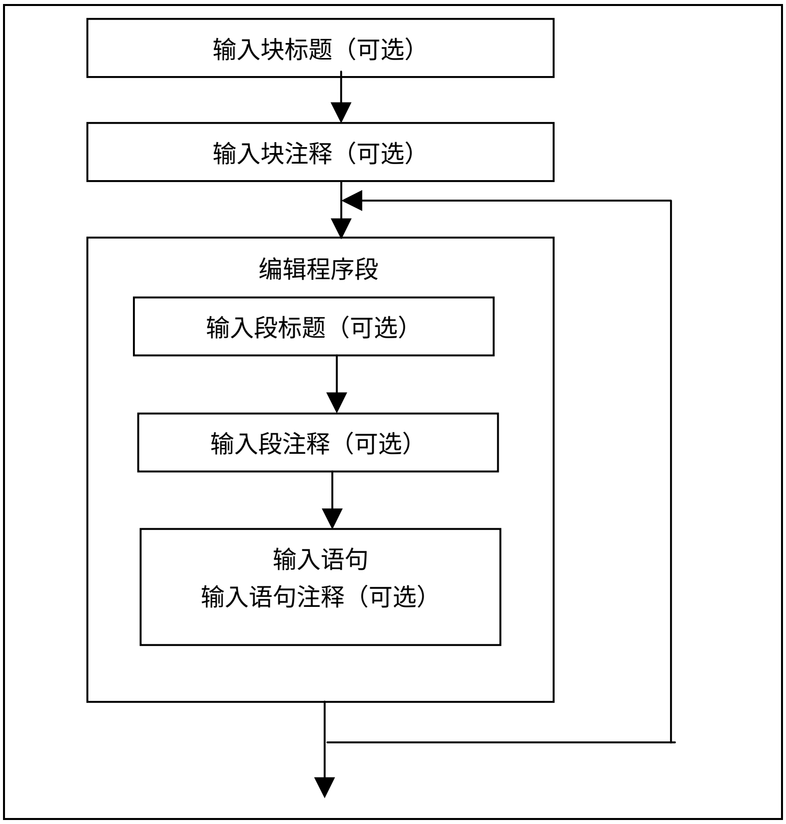
在程序指令部分，用户可以编辑块标题，块注释，段标题，段注释和各程序段中的语句行。

用STL编程语言的程序指令部分结构



9.4.2 输入语句的步骤

用户可以按任意的顺序来编辑程序指令部分的各项内容。当你第一次编程时，我们建议你按如下步骤进行。



用户既可以在修改状态也可以在新插入状态修改上述内容。用执行键（INSERT）切换两种状态。

9.4.3 在程序中输入共享符号

用菜单命令 Insert>Symbol，用户可在程序的指令部分插入符号，当光标位于字符串的开始、结束或中间时，如果符号已经存在，则总是以小短线“ - ”开始。如果用户修改字符串，则该选择也在符号表中刷新。

一个字符串开始和结尾的分开标志是，例如，空格、句号、逗号。在共享符号中没有分开标志。

按如下步骤输入符号：

1. 在程序中输入所需符号的第一个字母。
2. 同时按下 CTRL 和 J 键，显示符号表。第一个符号按你所输入的字母开始。
3. 按 RETURN 键输入符号或选择另一个符号，

然后，有引号圈起来的符号代替第一个字母。

通常下列情况实现：如果光标位于一个字符串的开始、结尾或中间，当插入一个符号时，该字符串被带引号的符号所替代。

9.4.4 块和段的标题与注释

注释可使用程序易于阅读，因此，也使得程序调试和故障诊断容易进行且更有效率。它们是程序文档中的重要组成部分，确实应该加以利用。

在梯形图、功能块图和语句表程序中的文字注释

可用下列文字注释：

- 块标题：块的标题（最多64个字符）
- 块注释：整个逻辑块的文字说明，例如：该块的目的。
- 段标题：段的标题（最多64个字符）
- 段注释：单个段的功能说明。
- 变量声明表中的注释栏：所声明的本块数据的说明。
- 符号注释：在符号表中当符号名及地址声明之后的文字说明。

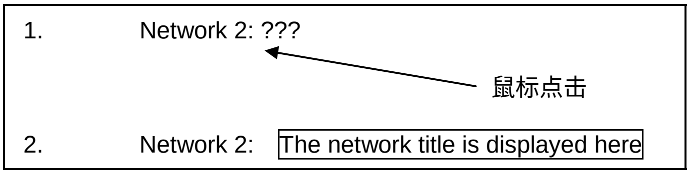
用户可以用菜单命令 View>Display with>Symbol Information，显示这些注释。

在逻辑块的程序部分，用户可以输入块标题、段标题、块注释或段注释。

块标题或段标题

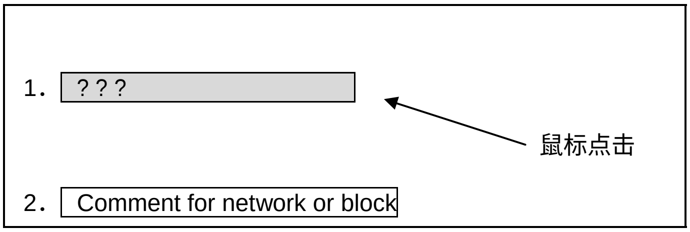
要想输入块标题或段标题，需把光标放在三个问号的位置，向右输入块名称或段名（例如 Network1：???）。当你输入标题时打开一个文字输入框，最长可输入 64 个字符。

块注释是整个逻辑块的注释，在块注释中，可以注释块的功能。段注释是具体段的注释，可以详细说明段。



块注释和段注释

用户可以用菜单命令 View>Display with>Comments，显示或不显示灰色的注释域。双击注释域打开文字输入框，可输入注释要点。当你输入标题时打开一个文字输入框，最长可输入 64 个字符。



9.4.5 在程序指令部分搜索错误的功能

在程序指令部分的错误，由于是红颜色，很容易识别。编辑器提供两种搜索功能 Edit>GoTo>Previous Error/Next Error，用于查找屏幕可见部分以外的错误。

搜索错误的范围超过一个程序段。这就意味着在整个程序指令部分搜索，不只是一个段，或当前屏幕能看见的范围。

如果用菜单命令 View>Status Bar 激活状态条，找到的错误显示在哪儿。

用户也可在修改状态下修改错误和进行改进。用执行键（INSERT）可切换插入状态和修改状态。

9.5 在程序指令部分中用梯形图（LAD 编程）

9.5.1 梯形逻辑编程的一些设置

设置梯形逻辑布局

用户可以设置在梯形逻辑表示法中生成程序的布局。所选的格式（A4 窄幅/宽幅/最大尺寸）将影响一行中所能显示的梯形元素的数目。

1. 选择菜单命令 Options > Customize。
2. 在出现的对话框中选择 “ LAD/FBD ” 页。
3. 在 “ Layout（布局） ” 列表框中，选择所需的格式。输入所需格式的大小。

设置打印

如果希望打印梯形程序指令部分，用户应在开始编程之前，设置合适的打印纸格式。

在 “ LAD/FBD ” 页中的设置

用菜单命令 Options>Customize，可访问 “ LAD/FBD ” 页，在该页中你可以做基本设置，例如布局和地址域宽。

9.5.2 输入梯形逻辑元素的规则

用户在《S7-300/400 梯形逻辑-编辑手册》一书中，或梯形逻辑在线帮助中，可以找到有关梯形逻辑编程语言表示法的描述。

一个梯形程序段中，可以有多个分支，每条分支上可有多个元素。所有的元素和分支都必须连接；左边的能源轨不算连接（IEC 1131-3）。

当用户编写梯形程序时，必须遵守编程规则。如果有错误发生，会有信息提示。

结束梯形程序段

每个梯形程序段都必须以输出线圈或功能框结束，下列的梯形元素不能用于程序段结束：

- 比较框
- 中间输出结果的线圈 _/(#)_/
- 上升沿 _/(P)_/ 或下降沿 _/(N)_/ 线圈

功能框的位置

用于功能框连接的分枝起始点必须总是左边的能源轨，在该功能框前的分支上可以有逻辑操作或其它功能框。

线圈的位置

线圈自动位于程序的最右端，在这个位置上形成分支的终点。

下列情况除外：中间输出结果的线圈 _/(#)_/和正极 _/(P)_/或负

用于中间结果输出的线圈_/(#)_/ ，以及上升沿线圈 _/(P)_/ 或下降沿线圈 _/(N)_/ ，都不能置于分支的最左端或最右端。也不允许放在平行分支上。

有些线圈要布尔逻辑操作，而有些线圈一定不能不能用布尔逻辑操作。

- 要求布尔逻辑的线圈为：
  - 输出 _/()，置位输入_/(S)，复位输入 _/(R)
  - 中间结果输出_/(#)_/，上升沿 _/(P)_/，下降沿 _/(N)_/
  - 所有的计数器和定时器线圈
  - 逻辑非跳转 _/(JMPN)
  - 主控继电器接通 _/(MCR<)
  - 将RLO存入BR存贮器_/(SAVE)
  - 返回_/(RET)
- 不允许布尔逻辑的线圈：
  - 主控继电器激活 _/(MCRA)
  - 主控继电器取消_/(MCRD)
  - 打开数据块_/(OPN)
  - 主控继电器关_/(MCR)

所有其它的线圈既可以用布尔逻辑操作也可以不用。

下列线圈一定不能用于平行输出：

- 逻辑非跳转 _/（JMPN）
- 跳转 _/（JMP）
- 从线圈调用 _/（CALL）
- 返回_/（RET）

使能输入/使能输出

功能框的使能输入端“ EN ”和使能输出端“ ENO ”可以连接使用，也可以不用。

删除和修改

如果分支中只有一个元素，当删除这个元素时，整个分支也同时删掉。

当一个功能框删除时，该功能框的所有布尔输入分支都将删除，主分支除外。

修改状态可用于简单的同类型元素的覆盖。

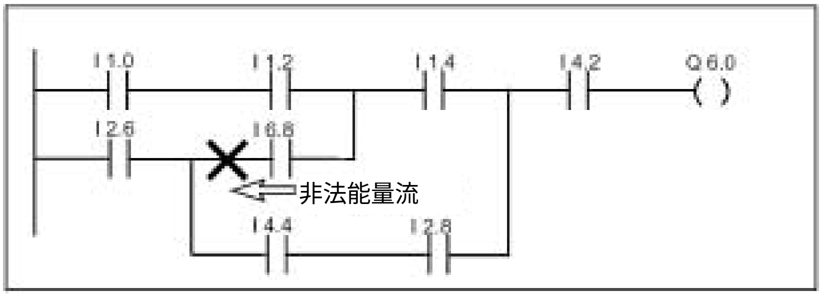
平行分支

- 从左到右画一个或（OR）分支。
- 向下打开平行分支，向上闭合。
- 选择某一梯形元素后，总是可以打开一个平行分支。
- 在选择的梯形元素之后，总是可以闭合一个平行分支。
- 删除一个平行分支，将删掉该分支上的所有元素。当分支上最后一个元素删除，则分支自动删除。

9.5.3 梯形图中的非法逻辑操作

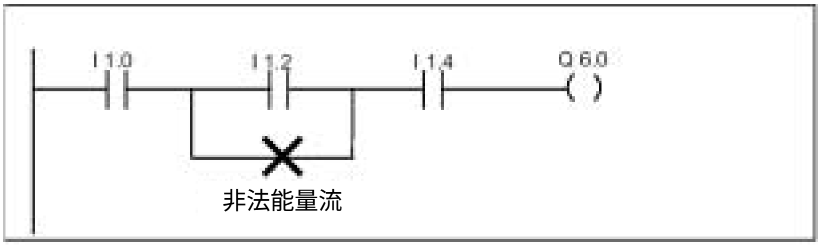
能量流从右到左

不允许生成使能量流向相反方向的分支。下图所示为：当 I 1.4 的信号状态为“0”时，能量流经 I 6.8 的方向是从右到左。这是不允许的。



短路

不允许生成造成短路的分支。下图所示为举例：下图所示为：



9.6 在程序指令部分用功能块图（FBD）编程

9.6.1 功能块图编程的一些设置

设置功能块图布局

用户可以设置在功能块图表示法中生成程序的布局。所选的格式（A4 窄幅/宽幅/最大尺寸）将影响一行中所能显示的 FBD 元素的数目。

1. 选择菜单命令 Options > Customize。
2. 在出现的对话框中选择 “ LAD/FBD ” 页。
3. 在 “ Layout（布局） ” 列表框中，选择所需的格式。输入所需格式的大小。

设置打印

如果希望打印 FBD 的程序指令部分，用户应在开始编程之前，设置合适的打印纸格式。

在 “ LAD/FBD ” 页中的设置

- 用菜单命令Options>Customize可访问 “ LAD/FBD ” 页，在该页中你可以做基本设置，例如布局和地址域宽。

9.6.2 输入 FBD 元素的规则

用户在《S7-300/400 功能块图编程手册》一书中，或 FBD 在线帮助中，可以找到有关 FBD 编程语言表示法的描述。

一个 FBD 程序段中，可以有多个元素，所有的元素都必须连接（IEC1131-3）。

当用户编写 FBD 程序时，必须遵守编程规则，如果有错误发生，会有信息提示。如果有错误发生，会有信息提示。

输入和编辑地址和参数

当插入一个 FBD 元素时，符号“ ??? ”和“ … ”用作地址和参数的标记符号。

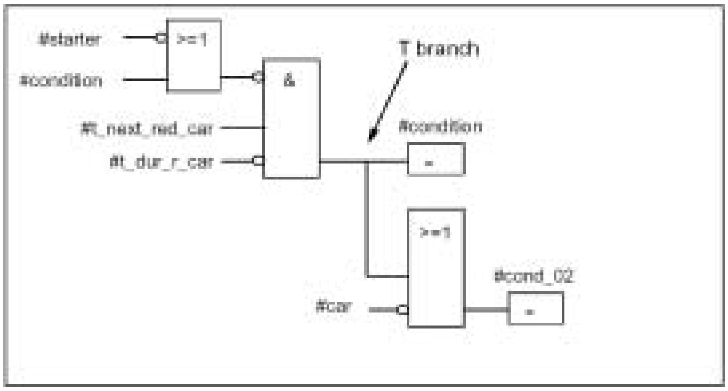
- 红色的标记符号“ ??? ”表示其地址和参数必须修改。
- 黑色的符号“ … ”表示其地址和参数可以修改

如果用户将鼠标点在标记符号上，可显示相应的数据类型。

功能框的位置

标准功能框（触发器、计数器、定时器、算术操作等）可插在二进制逻辑操作框（&（与），>=（或），XOR（异或））的中间。比较功能框除外。

在一个程序段内不允许有不同逻辑操作的不同输出的程序。可是，用户可以利用 T 型分支将一串逻辑操作分别赋值给不同的输出。下图所示为一个程序段中有两个赋值输出。



下列功能框只能放在逻辑串的最右边，在那结束该逻辑串。

- 设置计数器值
- 加计数器参数赋实参（实际参数），减计数器形参赋实参
- 脉冲定时器参数赋值及起动，扩展脉冲定时器参数赋值及起动。
- 接通延迟/断开延迟定时器的参数赋值和起动。

有些功能框要求布尔逻辑操作，而有些功能框一定不能用布尔逻辑操作。

要求布尔逻辑操作的功能框：

- 输出，置位输出，复位输出 _/[R]
- 中间结果输出_/[#]_/，上升沿_/[P]_/，下降沿_/[N]_/
- 所有的计数器和定时器功能框
- 逻辑非跳转 _/[JMPN]
- 主控继电器接通 _/[MCR<]
- 将RLO存入BR存贮器 _/[SAVE]
- 返回 _/[RET]

不允许布尔逻辑的功能框：

- 主控继电器激活[MCRA]
- 主控继电器取消[MCRD]
- 打开数据块[OPN]
- 主控继电器断开[MCR>]

所有其它功能框既可用布尔罗辑，也可不用。

使能输入/使能输出

功能框的使能输入端“ EN ”和使能输出端“ ENO ”可以连接使用，也可以不用。

删除和修改

当一个功能框删除时，该功能框的所有布尔输入分支都将删除，主分支除外。

修改状态可用于简单的同类型元素的覆盖。

9.7 在程序指令部分编辑 STL 语句

9.7.1 语句表编程的设置

记忆性设置

用户可以选择两种记忆性设置：

- German（德语）
- English（英语）

用户在打开逻辑块之前，在 SIMATIC 管理器中通过菜单命令 Options> Customize 中的“Language（语言）”页来设置记忆性。在编辑块时，不能改变记忆码。

在对话框中，编辑块属性。

在编辑器中，用户可以打开许多块，需要时交替地进行编辑。

9.7.2 输入 STL 语句的规则

用户在《S7-300/400 语句表编程手册》一书中，或 STL 在线帮助中，可以找到有关语句表编程语言的描述。

当用户在增量输入方式下输入 STL 语句时，必须注意以下基本规则：

- 逻辑块编程的顺序很重要。在调用某逻辑块之前，该块必须已编好并存在。
- 每条语句都是由标号（可选），指令，地址和文字注释（可选）组成。
  例如： M001：A I 1.0 //Comment
- 每条语句占一行。
- 逻辑块中最多有999个程序段
- 每个程序段最多将近2000行左右。如果用户使用放大或缩小功能，相应的行数多些或少些。
- 当输入指令或绝对地址时，不区分大小写。

9.8 刷新块调用

9.8.1 刷新块调用

用户可以在“ LAD/STL/FBD—Programming S7 Blocks ”编辑器中，用菜单命令 Edit> Block Call>Update，自动刷新那些由于下列接口发生变化而变为非法的块调用或用户定义数据类型：

- 插入新参数
- 删改参数
- 修改参数名
- 修改参数类型
- 修改参数的顺序

当用户进行形式参数和实际参数的赋值时，必须按顺序遵循下列规则：

1. 参数名相同：
   如果形参名保持不变，实参自动赋值。
   特例：在梯形图和功能块图中，预先连接的二进制输入参数只有将数据类型（BOOL）相同时才自动赋值。如果数据类型改变，则预先连接变为开路分支。

2. 参数的数据类型相同：
   当同名参数已经赋值完成之后，迄今还未赋值的实参将赋值给那些与“ 旧 ”形参数据类型相同的形参。

3. 参数位置相同：
   当用户按规则1和2执行完之后，还没有赋值的那些实参，现在按照他们在“ 旧 ”接口的参数位置，赋给形参。

4. 如果按照上面三条规则，还有实参不能赋值，它们将被删除，或在梯形图和功能图中预先连接的二进制信号，将变为开路分支。

当执行完这个功能之后，请检查在变量声明表和程序指令部分做的修改。

9.9 逻辑块存盘

9.9.1 逻辑块存盘

输入新生成的程序块或在编程器数据库中修改了程序指令部分或变量声明表之后，用户必须将相应的块存盘。然后，该数据写到编程器的硬盘中。

将逻辑块存入到编程器的硬盘中：

1. 激活你想存盘的块的工作窗口。
2. 选择下列菜单命令之一：
   - File>Save，将块存在同一名下。
   - File>Save As，将块存在不同的 S7 用户程序下或不同的名下。在出现的对话框中输入新的路径或新的块名。

在以上两种情况中，只有当逻辑块中没有句法错误时才能存盘。当生成逻辑块时，出现句法错误，会立即识别并用红颜色显示出来。这些错误必须在存盘之前修改。

注意

- 用户也可以在SIMATIC管理器中（比如，用拖放功能）将逻辑块或源文件存到其它项目或程序库中。
- 用户在SIMATIC管理器中，只能将块或整个用户程序存到存储卡中。
- 如果某些较大的逻辑块在存盘或编译时出现问题，用户应重新组织项目。在SIMATIC管理器中，用菜单命令File > Reorganize 来进行。然后，再试一次存盘或编译。

9.9.2 在功能、功能块或 UDT 中修改接口

如果用户需要修改 FB、FC 或 UDT 中的接口，按如下步骤，以避免产生时间标记冲突：

1. 用想修改的块以及跟该块直接或间接相关的块产生一个 STL 源文件。
2. 在用户产生的源文件中将修改后的程序存盘。
3. 再将修改后的源文件编译回逻辑块。

现在用户就可以存盘/下载修改接口的块了。

9.9.3 块调用时避免出错

STEP 7覆盖DB寄存器中的数据

当各种指令执行时，STEP 7 将修改 S7-300/S7-400 CPU 的寄存器。例如：当用户调用一个 FB 时，DB 寄存器和 DI 寄存器中的内容进行交换。这使得在所调 FB 的背景数据块打开的同时，不会选择前一个背景 DB 的地址。

如果用户使用绝对寻址，当访问存在寄存器中的数据时，会出错误。在一些情况下，在寄存器 AR1（地址寄存器 1）中的地址和在 DB 寄存器中的地址被覆盖。这意味着，你可能读或写的是错误地址。

危险

当如下操作进行时，会有破坏属性和人身的危险：

1. 用 CALL FC，CALL FB，CALL 多重背景
2. 全部用绝对地址（例如：DB20.DBW10）访问 DB。
3. 复杂数据类型的变量访问。

DB 寄存器（DB 和 DI）、地址寄存器（AR1，AR2）和累加器（ACCU1，ACCU2）的内容可能已被改变。

此外，当用户调一个 FB 或 FC 时，不能用状态字的 RLO 位作为一个额外的（隐含的）参数。

当使用上述编程技巧时，用户必须确保自己保护并重装那些内容，否则会出错。

保护正确的数据

如果用户用缩写的格式访问数据的绝对地址，则 DB 寄存器的内容会造成危险情况。例如 :如果假设 DB20 是打开的(其号码存在 DB 寄存器中)，用户可用 DBX0.2 访问地址在 DB 寄存器中的 DB（即：DB20）的 0 字节的 bit2 的数据。可是，如果 DB 寄存器中的 DB 号码不对，则将访问一个错误的数据。

用下列方法访问数据，可避免访问 DB 寄存器的数据时出错：

- 用符号地址
- 用完整绝对地址（例如：DB20.DBX0.2）

如果用这些寻址方法，STEP 7 自动打开正确的 DB。如果用 AR1（寄存器）进行间接寻址，用户必须在 AR1 中装入正确的地址。

寄存器被修改的情况

地址寄存器用于间接寻址的操作只能用 STL 形式。其它语言不支持间接访问地址寄存器。

在所有的编程语言中，必须考虑到编译器会对 DB 寄存器进行修改，以确保当块调用时传送正确的参数。

在下列情况中，所调块的地址寄存器 AR1 和 DB 寄存器中的内容被覆盖：

情　况	描　述
从 DB 中来的实参	• 一旦用户赋值一个从 DB 中来的实参（如：DB20.DBX 0.2），STEP 7 打开 DB（DB20）并修改 DB 寄存器中的内容，当块调用结束后，程序工作在修改了 DB 的情况下
当所调块使用高级数据类型时	• 当在一个 FC 中调用某块时，若使用高级数据类型（字符串），数组，结构或 UDT 传给该块的形参，则调用块（FC）的 ARI 和 DB 寄存器中的内容被修改 • 在 FB 中，如果调用块的形参类型在 VAR_IN_OUT 区，则效果与上述相同
当访问高级数据类型的元素时	• 当 FB 在 VAR_IN_OUT 区访问高级数据类型的形参元素（字符串、数组、结构或 UDT）时，STEP 7 将使用地址寄存器 AR1 和 DB 寄存器。这就意味着这两种寄存器的内容将被修改

注意

- 当FB是在版本1中的块里调用时，如果在调用指令之前没有限制RLO，则第一个二进制的布尔型IN或IN_OUT形参不能正确地进行实参传送。在这种情况下，会将已存在的RLO逻辑地结合一起。
- 当调用某一FB时（单一背景或多重背景），地址寄存器AR2要改写。
- 如果在一个FB中，地址寄存器AR2被修改了，就不能保证该FB能正确地执行。
- 如果整个DB的绝对地址没有传送到ANY参数，ANY指针就不会得到打开DB的DB编号。相反，它会得到数字“ 0 ”。

10 数据块的生成

10.1 有关生成数据块的基本信息

数据块（DB）可用于，例如，存贮设备或生产线中可访问的值。与用编程语言梯形图、语句表或功能块图之一编程的逻辑块不同，数据块只有变量声明部分。这就意味着，在这里不用程序指令部分，所以也就没有程序段。

当用户打开数据块时，既可用声明表形式显示，也可用数据显示形式浏览数据块。用户可以用菜单命令 View>Declaration View 和 View>Data View，来切换两种显示状态。

声明表显示状态

如果用户有如下期望，应用声明表显示状态：

- 浏览或确定共享数据块的数据结构。
- 浏览带某相关的用户定义数据类型（UDT）的数据块的数据结构。
- 浏览与功能块（FB）相关的数据块的数据结构。

与功能块或用户定义数据类型相关的数据块的结构不能修改。要想修改它们，首先必须修改相应的 FB 或 UDT，然后再生成一个新数据块。

数据显示状态

如果用户想修改数据，就应用数据显示状态。在数据显示状态中，用户只能显示，输入或改变每个元素的实际值。在数据块的数据浏览中，复杂数据类型变量的各元素，分别用其全名列出。

背景数据块与共享数据块之间的区别

共享数据块不附属于任何逻辑块。它含有生产线或设备所需的值，并可以在程序的任何点直接使用。

背景数据块直接附属于某逻辑块，例如：功能块。背景数据块中所含数据为功能块的变量声明表中所存数据。

10.2 数据块的声明表显示状态

对于不能共享的数据块，在声明表显示状态不能进行修改。

列	解　　释
地址（Address）	当用户结束变量声明的输入时，STEP 7 自动分配并显示进址
声明类型　（Declaration）	本栏只在背景数据块中显示。表明在功能块的变量声明表中各变量是如何声明的： • 输入参数（IN） • 输出参数（IN） • 输入/输出参数（IN_OUT） • 静态数据（STAT）
名称　（Name）	这里输入给每个变量的符号名
数据类型　（Type）	输入用户赋给变量的数据类型（BOOL，INT，WORD，ARRAY，等）。变量可以有基本数据类型，复杂数据类型，或用户声明数据类型
初值　（Initial Value）	这里可输入初值，如果用户不想输入，则软件根据所输入的数据类型给出缺省值。 如果用户没有给变量声明实际值，当数据块第一次存盘时，初值将用做实际值
注释　（Comment）	输入对变量文档有帮助的注释，最多可以有 80 个字符

10.3 数据块中的数据总览

数据显示状态为用户显示该数据块中所有变量的实际值。用户只能在数据显示状态形式中改变这些值。对所有的共享数据块来说，数据显示状态的表格表示都是相同的。对于背景数据块，还增加一个“ 参数声明（Declaration） ”栏。

对于用复杂数据类型或用户定义数据类型的变量，在数据浏览时，所有的元素都占有自己的一行，并有其完整的符号名。如果元素位于背景数据块的 IN_OUT 区，则指针指向“ 实际值（Actual value） ”栏中的复杂数据类型或用户定义数据类型。

数据显示状态中有下列栏目显示：

栏	解　　释
地址（Address）	STEP 7 自动为变量分配并显示地址
声明类型 （Declaration）	本栏只在背景数据块中显示，表明在功能块的变量声明表中各变量是如何声明的： • 输入参数（IN） • 输入参数（IN） • 输入/输出参数（IN_OUT） • 静态数据（STAT）
名称（Name）	在变量声明时给的符号名。在数据浏览中用户不能编辑该域
数据类型（Type）	显示变量所声明的数据类型 对于共享数据块，这里只列出基本数据类型，因为在数据浏览中，复杂数据类型变量或用户声明数据类型要分别列出元素。 对于背景数据块，也显示参数类型，对于用复杂或用户声明数据类型的 in/out 参数（IN_OUT），指针指向“实际值”栏中的数据类型
初值（Initial Value）	用户为变量输入的初值，如果用户不想输入，则软件会根据所声明的数据类型给出缺省值。 如果用户没有给变量声明实际值，当数据块第一次存盘时，初值将用做实际值
实际数值 （Actual Value）	离线：打开数据块时或最后改变或保存的变量数值（即使是在线打开的数据块，也不刷新该显示）。 在线：将显示打开数据块时的实际数值，但不会自动刷新。按动 F5，可以刷新该窗口。 如果不属于复杂或用户定义数据类型的输入/输出参数（IN_OUT），你可以编辑该区域。所有的值必须与数据类型匹配。
注释 （Comment）	对变量进行说明所输入的注释。在数据浏览中用户不能编辑该域。

10.4 数据块的编辑与存盘

10.4.1 输入共享数据块的数据结构

如果用户打开一个不是用功能块或用户定义数据类型生成的数据块，则可以在该数据块的声明表显示状态中声明其结构。对于不能共享的数据块，在声明表显示状态不能进行修改。

打开一个共享数据块，意为该数据块与 UDT 或 FB 无关。

如果声明表显示状态没有设置，则需要选择该数据块的声明表显示状态进行显示。

按下列信息，填写表格来声明数据块的结构。

列	解释
地址 （Address）	当用户结束变量声明的输入时，STEP 7 自动分配并显示地址
名称（Name）	这里输入给每个变量的符号名
数据类型 （Type）	输入用户赋给变量的数据类型（BOOL，INT，WORD，ARRAY，等）。变量可以有基本数据类型，复杂数据类型，或用户声明数据类型
初值 （Initial Value）	这里可输入初值，如果用户不想输入，则软件根据所输入的数据类型给出缺省值。 如果用户没有给变量声明实际值，当数据块第一次存盘时，初值将用做实际值
注释 （Comment）	输入对变量文档有帮助的注释。最多可以有 80 个字符。

对于不能共享的数据块，声明表显示状态不能修改。

10.4.2 输入并显示与 FB 有关的数据块（背景 DB 的数据结构）

输入

当用户将数据块与某一功能块相连时（背景 DB），则功能块的变量声明表决定该数据块的结构。任何修改只能在相关的功能块中进行。

1. 打开相关的功能块（FB）。
2. 编辑该功能块的变量声明表。
3. 再生成背景数据块。

显示

在背景数据块的声明表显示状态中，用户可以显示功能块的变量是如何声明的。

1. 打开数据块。
2. 如果声明表显示状态没有设置，则需要选择该数据块的声明表显示状态进行显示。
3. 参看下表中有关声明表的更多信息。

对于不能共享的数据块，在声明表显示状态不能进行修改。

列	解释
地址（Address）	STEP 7 自动为变量分配并显示地址。
参数类型 （Declaration）	本栏表明在功能块的变量声明表中各变量是如何声明的： • 输入参数（IN） • 输入参数（IN） • 输入/输出参数（IN_OUT） • 静态数据（STAT） 所声明的功能块临时局域数据不位于背景数据块中。
名称（Name）	在功能块变量声明表中给出的符号名
数据类型 （Type）	显示功能块的变量声明表中给出的数据类型。变量可以有基本数据类型，复杂数据类型，或用户声明数据类型。 如果在功能块中调用其它功能块，必须声明其调用静态变量，用这个静态数据类型也可声明一个功能块或一个系统功能块（SFB）

列	解释
初值 （Initial Value）	用户在功能块的变量声明表中输入初值，如果不想输入，则软件给出缺省值。 当数据块第一次存盘时若用户没有明确地声明实际值，则初值将被用于实际值。
注释（Comment）	在功能块的变量声明表中输入的注释，以便对数据元素文字说明。用户不能编辑该域。

注意

对于分配给某功能块的数据块，用户只能编辑变量的实际值。为输入变量的实际值，用户必须工作在数据块的数据浏览形式中。

10.4.3 输入用户定义的数据类型（UDT）的数据结构

1. 打开用户定义数据类型（UDT）。
2. 显示声明表形式如果该显示形式没设置的话。
3. 通过声明变量的顺序，变量的数据类型以及初值确定 UDT 的结构。下表中给出所需的信息。
4. 用 TAB 或 RETURN 键退出某行，来完成该行的变量输入。

列	解　释
地址（Address）	当用户结束变量声明的输入时，STEP 7 自动分配并显示地址。
名称（Name）	这里输入给每个变量的符号名。
数据类型（Type）	输入用户赋给变量的数据类型（BOOL，INT，WORD，ARRAY，等）。变量可以有基本数据类型，复杂数据类型，或变量自己的用户声明数据类型。
初值（Initial Value）	这里可输入初值，如果用户不想输入，则软件根据所输入的数据类型给出缺省值。所有的值必须与数据类型匹配。如果用户没有给变量声明实际值，当第一次用户定义数据类型或一个变量，或一个数据块的背景存盘时，初值将用做实际值。
注释（Comment）	输入对变量进行文字说明的注释，最多可以有 80 个字符。

10.4.4 输入并显示与 UDT 相关的数据块结构

输入

当用户用 UDT 生成某数据块时，UDT 的数据结构就决定了该数据块的结构。任何修改都只能在相关的用户定义数据类型（UDT）中进行。

1. 打开用户定义数据类型（UDT）。
2. 编辑用户定义数据类型的结构。
3. 再生成数据块。

显示

用户只能在数据块的声明表显示状态中，显示用户定义数据类型中变量是怎样声明的。

1. 打开数据块。
2. 如果声明表显示状态没有设置，则需要选择该数据块的声明表显示状态进行显示。
3. 参看下表中有关声明表的更多信息。

在声明表显示状态下不能进行修改。任何修改都只能在相关的用户定义数据类型（UDT）中进行。

列	解　　释
地址（Address）	STEP 7 自动为变量分配并显示地址
名称（Name）	在用户数据类型的变量声明表中给出的符号名
数据类型（Type）	显示在用户声明数据类型的变量声明表中给出的数据类型。变量可以有基本数据类型，复杂数据类型，或用户声明数据类型。
初值（Initial Value）	用户在用户声明数据类型中为变量输入的初值，如果用户不想输入，则软件给出缺省值。 当数据块第一次存盘时若用户没有明确地声明实际值，则初值将被用于实际值。
注释（Comment）	在用户声明数据类型的变量声明表中输入的注释，用于对数据元素进行文字说明

注意

对于用 UDT 生成的数据块，用户只能编辑变量的实际值。为输入变量的实际值，用户必须工作在数据块的数据浏览形式中。

10.4.5 在数据显示状态下编辑数据值

只有在数据块的数据显示状态下，才可能编辑实际值。

1. 如果需要，用菜单命令 View>Data View，将显示表格切换到数据显示状态。
2. 在“实际值（Actual Value）”一栏的域里输入各数据元素的实际值。实际值必须与这些数据元素的数据类型相匹配。

任何不正确的输入（例如，如果输入的实际值与数据类型不匹配，在编辑时能立即识别，并显示为红颜色）。这些错误必须在数据块存盘之前改正过来。

注意

一旦数据块存盘，数据值的任何变化都将保存。

10.4.6 将数据值恢复为初始值

只有在数据块的数据显示状态，才可能将数据值恢复为初始值。

1. 如果需要，用菜单命令 View>Data View，将显示表格切换到数据显示状态。
2. 选择菜单命令 Edit>Initialize Data Block 来进行。

所有的变量又恢复了初始值，这意味着所有变量的实际值被它们的初始值所替代。

注意

一旦数据块存盘，数据值的任何变化都将保存。

10.4.7 存储数据块

在编程器数据库中，输入的新生成的块或在数据块中修改了数据值，用户都必须将相应的块存盘。然后，该数据写到编程器的硬盘中。

将逻辑块存入到编程器的硬盘中：

1. 激活你想存盘的块的工作窗口。
2. 选择下列菜单命令之一：
   - File>Save，将块存在同一名下。
   - File>Save as，将块存在不同的 S7 用户程序中或不同的名下。在出现的对话框中输入新的路径或新的块名。对于数据块，用户不能使用 DB0，因为这个号码被系统占用。

在以上两种情况中，只有当逻辑块中没有句法错误时才能存盘。当生成逻辑块时，出现句法错误，会立即识别并用红颜色显示出来。这些错误必须在存盘之前修改。

注意

- 用户也可以在SIMATIC管理器中（比如，用拖放功能将逻辑块或源文件存到其它项目或程序库中）。
- 用户在SIMATIC管理器中，只能将块或整个用户程序存到存储卡中。
- 如果某些较大的逻辑块在存盘或编译时出现问题，用户应重新组织项目。在SIMATIC管理器中，用菜单命令File > Reorganize 来进行。然后，再试一次存盘或编译。

---

11 建立 STL 源文件

11.1 编写 STL 源文件的基本信息

用 STL 源文件输入程序然后将其编译成实用的软件块。这类源文件包括一个代码，这个代码与编译后的软件块号码相同。

用源文件形式编写程序有如下优势：

- 用ASCII编辑器编写源文件，而后将其编译成实用软件块。并将这些软件块存贮在S7用户程序文件夹中。
- 用源文件编写软件块的号码。
- 源文件可保存，即使含有语句错误。如果使用对逻辑软件块语法检测，则不能保存。因为你一旦编译源文件其语句错误就会暴露出来。

用语句表（STL）方式编写源文件。并设定它的软件块结构、变量表和程序段关键字。

当建立 STL 源文件时，应注意以下几点：

- 编写STL源文件的有关注意。
- 软件块在STL源文件中的语句及格式。
- 软件块在STL源文件中的结构方式。

11.2 编写 STL 源文件的规则

11.2.1 在 STL 源文件输入语句的规则

STL 源文件由文本方式组成。为将其编译成软件块，必须按一定规则编写其结构及语句。

请注意下列有关建立 STL 源文件的注意：

标　题	规　则
语句	除 CALL 指令结构以外，与 STL 语句表中的语句相同。
CALL	在源文件中将参数输入到括号内，并将各参数用逗号分开。 例如：FC call（one line） CALL FC10 （param1：=I0.0，param2：=I0.1）； 例如：FB call （one line） CALL FB10， DB100 （para1：=I0.0，para2：=I0.1）； 例如：FB call （more than one line） CALL FB10， DB100 （ para1：=I0.0， para2：=I0.1）； 注意： 调用软件块时，应按 ASCII 码编辑器依次编写的参数顺序完成参数的转换。否则 STL 与源文件中的赋值就会不匹配。
大/小写	除系统属性和转移选项卡以外，编辑器不能识别大小写。当输入字符串时（数据类型 STRING），必须确定大小写。关键字要用大写字母表示。在编译的时候看不到大小写，因此可以用大写、小写或大小写混合方式输入关键字。
分号	在 STL 语句和变量表的结尾使用分号（；）。这样就可以在一行输入多条语句。
双斜线（// ）	双斜线（// ）作为注释的开始，RETURN 作为注释的结束。

11.2.2 在源文件中声明变量的规则

源文件中的每一个软件块都要声明所需的变量。

变量声明表应设置在每个软件块的代码部分之前。

如果某些变量正在被调用，应按声明类型以正确的顺序声明这些变量。也就是说这类变量是一体的。

以梯形图、功能块图和语句表方式编程时，虽然填写了变量声明表，但是还要设置相关的关键字。

变量声明表的关键字

声明类型	关键字	用于
输入参数	″VAR_INPUT″ 声明表 ″END_VAR″	FB，FC
输出参数	″VAR_OUTPUT″ 声明表″ END_VAR″	FB，FC
输入/输出参数	″VAR_IN_OUT″ 声明表″ END_VAR″	FB，FC
静态变量	″VAR″ 声明表 ″END_VAR″	FB
临时变量	″VAR_TEMP″ 声明表″ END_VAR″	OB，FB，FC

关键字 END-VAR 表示声明表结束。

声明表是以某声明类型组成的变量表，表中的变量都有一个预置的值（VAR-TEMP 除外）。下面是一个声明表的结构的例子：

Duration_Motor1：　S5TIME　：=　S5T#1H_30M；

Variable　Data type　Default value

注意：

- 变量符号必须以字母开始。变量符号不能与已保留的关键字相同。
- 如果局部声明表与符号表中的变量符号相同，则应在局部声明表中的变量名前增加“ # ”符号，在符号表中的变量应加上引号。否则，此变量视为局部变量。

11.2.3 STL 源文件中软件块次序的规则

被调用的软件块应在调用它的软件块之前。意思是：

- 在大多数情况下，OB1用于调用其它的软件块，所以被调用的软件块应在OB1之前输入，OB1应在最后。
- 用户定义的数据类型（UDT）应在使用它的软件块之前输入。
- 带有用户定义的数据类型（UDT）的数据块应紧接在用户定义的数据类型之后。
- 如多个软件块使用同一数据块，则应首先输入这个数据块。
- 背景数据块应在其相应的功能块之后。
- DB0备用。用户不能建立DB0。

11.2.4 STL 源文件设定系统属性的规则

设定作用于软件块和参数的系统属性。系统属性用于管理报文组态、连接组态、操作员接口功能及过程控制组态等功能。

- 在源文件中输入系统属性时有如下规则：
- 系统属性的关键字以“ S7_ ”开始
- 系统属性应位于括号内。
- 句子格式：{S7_idenifier：= 'string'}
- 标识代码应用“ ； ”分开。
- 用于软件块的系统属性应在软件块属性之前，在关键字ORGANIZATION_和TITLE之后。
- 用于参数的系统属性应与参数声明在一起，并且位于数据声明的冒号之前。大小写字符不同。在输入时大小写应有所区别。

在“ System Attributes ”下面，用菜单命令 File > Properties，可以检查或修改用于软件块的“ 属性 ”选项卡。

用菜单命令 File>Object Properties，检查或修改用于参数的系统属性，光标必须放在参数声明的名称上。

11.2.5 STL 源文件设定软件块属性的规则

块属性可使用户更容易辨识所生成的各程序块，并且还可以对这些程序块加以保护，防止非法修改。

在“ General Part1 ”和“ General Part2 ”选项卡下，用菜单命令 File>Properties，可以检查或修改软件块属性。

其它软件块属性依次在源文件输入。

请按下列规则在源文件中输入：

- 软件块属性应在变量声明表之前
- 每个软件块属性各占一行
- 各行用分号结束
- 用关键字指定软件块属性
- 输入的软件块属性在软件块属性表中顺序显示
- 对应各种软件块类型的有效的软件块属性列表：从软件块属性到软件块类型

注意：

块属性能够在 SIMATIC 管理器目标中显示并能在此编辑下列属性：AUTHOR、FAMILY、NAME 和 VERSION。

软件块属性和软件块次序

当输入块属性时，输入顺序如下表。

次序	关键字/属性	含义	举例
1.	[KNOW_HOW_PROTECT]	块保护；使用该指令后，当该块进行编译时，使程序部分不可见	KNOW_HOW_PROTECT
2.	[AUTHOR：]	作者名：公司名，部门名或其它名你（最多为没有空格的 8 个字符）	AUTHOR：Siemens，没有关键字
3.	[FAMILY：]	块系列名：例如，控制器（最多为没有空格的 8 个字符）	FAMILY：Controller，没有关键字

次序	关键字/属性	含义	举例
4.	[NAME：]	块名（最多 8 个字符）	NAME：PID， 没有关键字
5.	[VERSION：int1. int2]	块的版本号（两个数的范围均在 0 到 15 之间，意为 0.0 到 15.15）	VERSION：3.10
6.	[CODE_VERSION1]	功能块能否有多重背景的标识符。如果想用多重背景则功能块不应有该属性。	CODE_VERSION1
7.	[UNLINKED] 只适用于 DB	某个数据块的属性是“UNLINKED”，则不将其连接到程序中。	
8.	[READ_ONLY] 只适用于 DB	数据块的写保护；其数据只能读，不能修改。	FAMILY=Examples VERSION=3.10 READ_ONLY

11.2.6 每个软件块类型的许可软件块属性

下表所示为哪种块类型可具有哪些块属性：

特性	OB	FB	FC	DB	UDT
KNOW_HOW_PROTECT	•	•	•	•	-
AUTHOR	•	•	•	•	-
FAMILY	•	•	•	•	-
NAME	•	•	•	•	-
VERSION	•	•	•	•	-
UNLINKED	-	-	-	•	-
READ_ONLY	-	-	-	•	-

使用KNOW_HOW_PROTECT设定块保护

当用户在 STL 源文件中编程块时，通过使用关键字“KNOW_HOW_PROTECT”设定块保护，可以保护块，防止非特许用户使用。

该块保护命令具有下列效果：

- 如果想在增量STL、FBD或梯形图编辑器中输出编译后的保护块，则该块的程序部分不能显示。
- 该块的变量声明表中只显示类型为 var_in、var_out和var_in_out的变量。类型为var_stat和var_temp的变量隐含。
- 在输入任何其它块属性之前，输入关键字KNOW_HOW_PROTECT。

使用READ_ONLY，设定数据块为写保护

对于数据块，用户可以设定写保护，以防在程序执行期间改写数据块。为此，数据块必须以 STL 源文件的形式存在。

使用源文件中的关键字“ READ_ONLY ”，可以设置写保护。这个关键字必须输入在声明变量之前的那一行。

11.3 软件块在 STL 源文件中的结构方式

11.3.1 软件块在 STL 源文件中的结构方式

在 STL 源文件中用关键字设定软件块结构。根据软件块的类型有如下结构选择：

- 逻辑块
- 数据块
- 用户定义的数据类型（UDT）

11.3.2 逻辑块在 STL 源文件中的结构方式

逻辑块由下列部分组成，各部分由相应的关键字定义：

- 块的开始部分
- 由关键字定义块的号码和名称，例如：
  - “ ORGANIZATION_BLOCK OB1 ” 定义组织块
  - “ FUNCTION_BLOCK FB6 ” 定义功能块，或
  - “ FUNCTION FC1：INT ” 定义功能。这样也同时设定了功能的类型。

  它可以是简单或复杂的数据类型（ARRAY 和 STRUCT 除外），并定义返回值的数据类型（RET_VAL）。如果没有返回值，则使用关键字“ VOID ”。
- 关键字“ TITLE ”引导软件块的标题（标题最长64个字符）。
- 用双斜线作为注释的开始
- 软件块的属性（可选）
- 变量声明表

- “ BEGIN ” 作为代码部分的开始。代码部分由1个或多个段组成，仅用“ NETWORK ” 作为段的标识符。不能输入段的号码。
- 关键字 “ TITLE= ” 作为段的开始。（标题最长64个字符）
- 用双斜线 “ // ” 作为每个段注释的开始。
- END_ORGANIZATION_BLOCK ， END_FUNCTION_BLOCK 或 END_FUNCTION作为软件块的结束。
- 软件块类型和号码之间应加空格。符号块名应加上引号，以确保局部变量符号名与符号表中的名称一致。

11.3.3 数据块在 STL 源文件中的结构方式

数据块由下列部分组成，各部分由自己的关键字引导：

- 启动部分：关键字和块号码或块名称，例如：DATA_BLOCK DB26
- 依照相关的UDT或功能块（可选）
- 由关键字 “ TITLE： ” 引导块的标题。（最长64个字符）
- 用双斜线 “ // ” 作为块注释的开始。
- 软件块的属性（可选）
- 变量声明表（可选）
- BEGIN作为缺省设置赋值部分的开始（可选）
- END_DATA_BLOCK作为块的结束

数据块有三种类型：

- 用户定义的数据块
- 带有用户定义的数据类型（UDT）的数据块
- 与功能块相联的数据块（作为背景数据块）

11.3.4 用户定义数据类型在 STL 源文件中的结构方式

用户定义的数据类型由下列部分组成，各部分由自己的关键字引导：

- 启动部分：关键字TYPE和块号码或块名称，例如：TYPE UDT20
- 结构化数据类型
- END_TYPE作为块的结束

当输入用户定义的数据类型时，数据类型应位于使用它的软件块之前。

11.4 软件块在 STL 源文件中的语句及格式

11.4.1 软件块在 STL 源文件中的语句及格式

如果编程 STL 源文件，格式表将显示语法和格式。语法说明如下：

- 各部分的说明在右边一栏。
- 输入内容应用引号。
- 方括号[...]其中内容可选择填写。
- 用大写字符输入关键字。

11.4.2 组织块的格式表

结　构	说　明
"ORGANIZATION_BLOCK" ob_no 或 ob_name	ob_no—块的符号，如 B1； ob_name—在符号表中定义的符号名
[TITLE=]	块标题（最长 64 个字符）
[块注释]	块注释应从“ //　”开始
[块的系统属性]	块的系统属性
[块的属性]	块的属性
变量声明表	临时变量声明表
"BEGIN"	关键字用于变量声明表和 STL 指令部分之间
NETWORK	段的开始
[TITLE=]	段标题（最多 64 个字符）
[段注释]	块注释应从“ //　”开始
STL 指令部分	块指令
"END_ORGANIZATION_BLOCK"	组织块结束的关键字

11.4.3 功能块的格式表

下表是 STL 源文件中功能块的简单格式表：

结　构	说　明
"FUNCTION_BLOCK" fb_no 或 fb_name	fb_no ：符号表中的符号名，如 FB6；fb_name 符号表中的符号名
[TITLE=]	块标题（最长 64 个字符）
[块注释]	块注释应从 " // " 开始
[块的系统属性]	块的系统属性
[块的属性]	块的属性
变量声明表	（输入、输出及输入/输出参数），临时变量，静态变量的声明表参数声明表包括参数的系统属性声明
"BEGIN"	关键字用于变量声明表和 STL 指令部分之间
NETWORK	段的开始
[TITLE=]	段标题（最多 64 个字符）
[段注释]	块注释应从 " // " 开始
STL 指令部分	块指令
"END_FUNCTION_BLOCK	表示功能块结束的关键字

11.4.4 功能格式表

下表是 STL 源文件中功能的简单格式表：

结　构	说　明
" FUNCTION " fc_no：fc_type 或 fc_name：fc_type	fc_no：为块编号，如 FC5； fc_name 为符号表中所定义的块符号名； fc_type：功能返回值（RET_VAL）的数据类型。它是一种简单或复杂的数据类型（ARRAY 和 STRUCT 除外）或 VOID。 如使用系统属性作为返回值（RET_VAL），必须将用于参数的系统属性输入在数据声明表之前。
[TITLE=]	块标题（最长 64 个字符）
[块注释]	块注释应从 " // " 开始
[块的系统属性]	块的系统属性

结　构	说　明
[块的属性]	块的属性
变量声明表	（输入、输出及输入/输出参数），临时变量，静态变量
"BEGIN"	关键字用于变量声明表和 STL 指令部分之间
NETWORK	段的开始
[TITLE=]	段标题（最多 64 个字符）
[段注释]	块注释应从 “ // ” 开始
STL 指令部分	块指令
“ END_FUNCTION ”	表示功能结束的关键字

11.4.5 数据块的格式表

下表是 STL 源文件中数据块的简单格式表：

结　构	说　明
"DATA_BLOCK" db_no 或 db_name	db_no：块的编号，如 DB5； db_name：符号表中的符号名
[TITLE=]	块标题（最长 64 个字符）
[块注释]	块注释应从 “ // ” 开始
[块的系统属性]	块的系统属性
[块的属性]	块的属性
声明部分	声明此块是否与 UDT 或 FB 相关，定义块的号码或符号表中的符号名或某种复杂的数据
"BEGIN"	关键字用于变量声明表和 STL 指令之间
[初始赋值]	变量具有初始值各变量或是赋值与常数值或赋与其它块相关的值
"END_DATA_BLOCK"	块结束的关键字

11.5 建立 STL 源文件

11.5.1 建立 STL 源文件

源文件必须在 S7 程序下的源文件夹中生成。可以在 SIMATIC 管理器或编辑器窗口生成源文件。

在 SIMATIC 管理器中生成源文件

1. 通过双击打开相应的 “ 源文件 ” 夹。
2. 要插入一个STL源文件，可以用菜单命令Insert>S7 Softwore>STL Source File。

在编辑器窗口生成源文件

1. 选择菜单命令 File＞New。
2. 在对话框中选择包含那些块的用户程序的同一个 S7 程序里的源文件夹。
3. 为新源文件输入一个名字。
4. 用 “ OK ” 确认。

该源文件在你输入的名字下生成，并显示一个编辑窗口。

11.5.2 编辑 S7 源文件

对于一个被编辑的源文件，可以在其对象特性中设置编程语言和编辑器。这就确保了当打开源文件编辑时，启动了正确的编辑器和正确的编程语言。STEP 7 标准软件支持在 STL 源文件中编程。

作为可选软件还有其它编程语言可以使用。如果相应的软件选项已装入你的计算机，你只能选择菜单命令插入源文件。

要编辑一个 S7 源文件可按如下进行：

1. 通过双击打开相应的 “ 源文件 ” 夹。
2. 按下述方法启动编辑所需的编辑器：
   - 双击右半窗口中所要的源文件。
   - 选择右半窗口中所需的源文件，并选择菜单命令 Edit>Open Object。

11.5.3 将软件块模板插入 STL 源文件

用于 STL 源文件编程的块模板有组织块（OB）、功能块（FB）、功能（FC）、数据块（DB）、背景数据块、与用户定义的数据类型相关的数据块、用户定义的数据类型（UDT）。块模板使得在源文件中输入块更容易，并且可以观察语法和结构框架。

要插入一个块模板可按如下进行：

1. 激活你要插入块模板的源文件窗口。
2. 将光标放在源文件中你想在其后插入块模板的位置。
3. 选择下面的菜单命令之一 Insert>Block Template>OB/FB/FC/DB / Instance DB/DB Referencing UDT/UDT。

块模板被插入到光标位置后的文件中。

11.5.4 插入外部源文件

可用 ASCII 编辑器生成并编辑源文件，然后将它导入到项目中，再用这个应用程序将它编译成块。要实现这一步，你必须将源文件导入到 S7 程序的“源文件”夹中，在编译过程中生成的块将存储在该 S7 程序的 S7 用户程序中。

要插入外部源文件可按如下进行：

1. 选择 S7 程序的源文件夹，外部源文件将被导入到该文件夹中。
2. 选择菜单命令 Insert>External Source File。
3. 在显示的对话框中选中要导入的源文件。

要导入的源文件必须有有效的扩展名。STEP 7 用扩展名来判定源文件的类型。这就意味着，例如，当 STEP 7 导入扩展名为.AWL 的文件时建立 STL 源文件。有效的文件扩展名列在对话框的“File Type（文件类型）”中。

注意

也可以用菜单命令 Insert>External Source File，导入你用 STEP 7 版本 1 生成的源文件。

11.5.5 从软件块建立 STL 源文件

你可以用已有的块生成一个可以在任何文本编辑器中编辑的 STL 源文件。该源文件生成在 S7 用户程序的源文件夹中，而所要选的块的用户程序也在同一个 S7 用户程序中。

要用块生成一个源文件可按如下进行：

1. 选择菜单命令 File>Generate Source File。
2. 在对话框中选择欲在其中生成新源文件的源文件夹。
3. 在文本框中为源文件输入名字。
4. 在“选择 STEP 7 块”对话框中，选择你想用来生成特定的源文件的块。所选的块显示在右边的列表框中。
5. 用“OK”确认。

来自所选块的 STL 源文件被生成并显示在编辑窗口中。

11.6 存储并编译 STL 源文件并执行一致性检查

11.6.1 存储 STL 源文件

你可以在任何时候在一个 STL 源文件的当前状态下对其进行存储。程序并不编译也不运行语法检查，这意味着任意错误都可以同时被存储。

语法错误的检查和报告只在源文件编译时或一致性检查后进行。

要在同一名下存储源文件：

1. 激活要存储的源文件的窗口。
2. 选择菜单命令 File>Save。

要将源文件存在一个新名/另一个项目下：

1. 激活要存储的源文件的窗口。
2. 选择菜单命令 File>Save As。
3. 在对话框中选择要在其中存储源文件的源文件夹并输入新名字。

11.6.2 检查 STL 源文件的一致性

用菜单命令 File>Consistency Check，可以显示源文件中的所有语法错误。与编译相比，不会有块生成。

当一致性检查完成后，有对话框显示已发现的错误的总数。

所有发现的错误分别列在窗口的下部并各有一行参考。为了能够生成各个块，在编译前修改这些错误。

11.6.3 在 STL 源文件中诊断故障

源文件的活动窗口被分割为两部分。在下半部分显示以下错误：

- 用菜单命令File>Compile，启动编译运行后所发现的错误。
- 用菜单命令File>Consistency Check，启动一致性检查后所发现的错误。

要找到错误在源文件中的位置，将光标放在窗口下部相应的错误信息上。窗口上部包含错误的文件行自动加亮。错误信息也显示在状态栏中。

11.6.4 编译 STL 源文件

要求

为了将你用 STL 源文件生成的程序编译成块，必须满足下列要求：

- 只有存储在S7程序下的 “ 源文件 ” 夹中的源文件才能被编译。
- 除了 “ 源文件 ” 夹之外，还要有一个 “ 块 ” 文件夹在S7程序下，编译过程中生成的块将存储在这个 “ 块 ” 文件中。只有源文件编译不出错。源文件中编程的块才能生成。如果在一个源文件中有许多块，只有那些没有错误的块才能生成。然后你就可以打开这些块编辑它们，下载CPU并一个一个地进行调试。

在编辑器中的编译过程

1． 打开要编译的源文件。该源文件必须在 S7 程序的源文件夹中，编译生成的块也要存储在这个 S7 程序的 S7 用户程序中。

2． 选择菜单命令 View>Display with>Symbolic Representation，以便在被编译的块中显示符号名。

3． 选择菜单命令 File>Compile。

4． “ 编译报告 ” 对话框显示出来，显示已编译的行和已发现的错误。

只有当源文件编译时无错才能生成文件指定的块。如果在一个源文件中有许多块，只有那些没有错误的块才能生成。错误警告不妨碍块的生成。

编译过程中查到的所有错误都显示在工作窗口的下部，并且在生成各个块之前必须进行改正。

在 SIMATIC 管理器中的编译过程

1. 通过双击打开合适的 “ 源文件 ” 夹。

2. 选择一个或多个要编译的源文件。对于一个关闭的源文件夹，无法启动编译运行来编译其包含的所有源文件。

3. 选择菜单命令 File>Compile，开始编译。你为源文件所选择的正确的编译器被调用。被成功编译的块则存储在 S7 程序的块文件夹中。

4. 编译过程中查出的所有语法错误显示在对话框中，并且必须被纠正，以便生成错误所在块。

11.7 STL 源文件的例子

11.7.1 STL 源文件声明变量的例子

基本类型的变量

```
                                    // 注释与声明表双斜线分开。
VAR_INPUT                           // 关键字：输入变量
         in1：INT；                 // 用“ ： ”将变量名和类型分开
         in3：DWORD；               // 变量声明用分号结束
         in2：INT：= 10；           // 设置声明表中的初始值
END_VAR                             // 同一类型的变量声明结束
VAR_OUTPUT                          // 关键字：输出变量
         out1WORD；
END_VAR                             // 关键字：临时变量
VAR_TEMP
         temp1：INT；
END_VAR
```

数组型的变量

```
VAR_INPUT                           // 输入变量
         array1：ARRA array1：ARRAY [1..20] of INT；  // 数组1是1维数组
         array2：ARRAY [1..20，1..40] of DWORD；      // 数组2是2维数组
END_VAR
```

结构型的变量

```
VAR_OUT                             // 输出变量
OUTPUT1：STRUCT                     // OUTPUT1具有STRUCT数据类型
         var1：BOOL；               // 结构中的元素1
         var2：DWORD；              // 结构中的元素2
         END_STRUCT；               // 结构结束
END_VAR
```

11.7.2 STL 源文件中组织块例子

```
ORGANIZATION_BLOCK OB1
TITLE = Example for OB1 with different block calls
// 第3段显示块的调用
// 带有或不带参数

{S7_m_c；= ’true’}            // 块的系统属性
AUTHOR            Siemens
FAMILY            Example
NAME              Test_OB
VERSION           1.1
VAR_TEMP
Interim value：INT；                 // 缓存器
END_VAR

BEGIN

NETWORK
TITLE = Function call transferring parameters
//   传送参数只有1行
CALL FC1 （param1：=I0.0，param2：=I0.1）；

NETWORK
TITLE = Function block call
// 传送参数
// 传送参数多行
CALL Traffic light control，DB6（            // FB名称，背景数据块
dur_g_p                 ：= S5T#10S，      //  参数赋实际值
del_r_p                 ：= S5T#30S，
starter                 ：= TRUE，
t_dur_y_car             ：= T 2，
t_dur_g_ped             ：= T 3，
t_delay_y_car           ：= T 4，
t_dur_r_car             ：= T 5，
t_next_red_car          ：= T 6，
```

```
r_car          ：= "re_main"，   // 引号表示符号
y_car          ：= "ye_main"，   // 在符号表输入名称
g_car          ：= "gr_main"，
r_ped          ：= "re_int"，
g_ped          ：= "gr_int"）；

NETWORK
TITLE = Function block call
// 传送参数
// 传送参数1行
CALL FB10， DB100（para1：=I0.0，para2：=I0.1）；
END_ORGANIZATION_BLOCK
```

11.7.3 STL 源文件中功能的例子

```
FUNCTION FC1：VOID
//  仅用于调用
VAR_INPUT
   param1：bool；
   param2：bool；
END_VAR
begin
end_function

FUNCTION FC2：INT
TITLE = Increment number of items
// 传送条目<1000个，这个功能增加了传送条目个数。
// 如果传送条目数超出1000个，
// 则在返回值功能（RET_VAL）显示“ -1 ”。

AUTHOR         Siemens
FAMILY         Throughput check
NAME：         INCR_ITEM_NOS
VERSION：      1.0
```

```
VAR_IN_OUT
ITEM_NOS：INT；     // 当前制造零件号
END_VAR

BEGIN

NETWORK
TITLE = Increment number of items by 1
// 运行的条目数少于1000个
// 计数器加1
L ITEM_NOS； L 1000；                    // 多于1个的例子
> I； JC ERR；                           // 行指令
L 0； T RET_VAL；
L ITEM_NOS； INC 1； T ITEM_NOS； BEU；
ERR：L -1；
T RET_VAL；
END_FUNCTION

FUNCTION FC3 {S7_pdiag：= 'true'}：INT
TITLE = Increment number of items
// 传送条目<1000个，这个功能
// 增加了传送条目个数。如果传送条目数超出1000个，
// 则在返回值功能（RET_VAL）显示“ -1 ”。
// RET_VAL具有参数的系统属性

AUTHOR：             Siemens
FAMILY：             Throughput check
NAME：               INCR_ITEM_NOS
VERSION：            1.0

VAR_IN_OUT
ITEM_NOS {S7_visible：= 'true'}：INT；     // 当前制造零件号
// 参数的系统属性
END_VAR
```

```
BEGIN

NETWORK
TITLE = Increment number of items by 1
// 运行的条目数少于1000个
// 计数器加1
L ITEM_NOS； L 1000；                   // 多于1个的例子
I； JC ERR；                            // 行指令
L 0； T RET_VAL；
L ITEM_NOS；INC 1；T ITEM_NOS；BEU；
ERR：L -1；
T RET_VAL；

END_FUNCTION
```

11.7.4 STL 源文件中功能块例子

```
FUNCTION_BLOCK FB6
TITLE = Simple traffic light switching
// 行人穿越
// 主路时指示灯的控制

{S7_m_c：= 'true'}       // 块的系统属性
AUTHOR        ：          Siemens
FAMILY        ：          Traffic light
NAME          ：          Traffic light01
VERSION       ：          1.3

VAR_INPUT

Starter       ：  BOOL：= FALSE；   //  对行人的要求
t_dur_y_car   ：  TIMER；          // 绿灯亮时对行人的要求
t_next_r_car  ：  TIMER；          // 红灯亮时对行人的要求
t_dur_r_car   ：  TIMER；
```

```
number      {S7_server：= ’alarm_archiv’； S7_a_type：= ’alarm_8’}：
            DWORD；
// 车的号码
// 号码是具有系统属性的参数

END_VAR
VAR_OUTPUT
g_car：BOOL：          =FALSE；     //   绿灯对车的要求

END_VAR
VAR
condition：BOOL：      =FALSE；     //   红灯对车的出现的条件
END_VAR

BEGIN
NETWORK
TITLE = Condition red for main street traffic
// 1分钟后，行人穿越主路的绿灯
// 作为红灯亮条件
// 主路交通量
          A（；
          A        #starter；             // 行人进路时绿灯的要求
          A        #t_next_r_car；        // 红灯亮的时间
          O        #condition；           // 或红灯亮的条件
          ）；
          AN       #t_dur_y_car；         // 红灯不亮的条件
          =        #condition；           // 红灯亮的条件

NETWORK
TITLE = Green light for main street traffic
          AN       #condition；           // 主街上没有红灯亮的条件
          =        #g_car；               //  主街上的绿灯

NETWORK
TITLE = Duration of yellow phase for cars
              // 增加控制
              // 交通灯的程序
```

```
END_FUNCTION_BLOCK

FUNCTION_BLOCK FB10
VAR_INPUT
para1：bool；
para2：bool；

end_var
begin
end_function_block

data_block db10
FB10
begin
end_data_block

data_block db6
FB6
begin
end_data_block
```

11.7.5 STL 源文件中的数据块举例

数据块

```
DATA_BLOCK DB10
TITLE = DB Example 10
STRUCT
        aa：BOOL；// Bool类型的变量
        bb：INT；// INT类型的变量INT
        cc：WORD；
END_STRUCT；

BEGIN           // 赋初值
            aa：= TRUE；
            bb：= 1500；
END_DATA_BLOCK
```

带有用户定义的数据类型的数据块

```
DATA_BLOCK DB20
TITLE = DB （UDT） Example
UDT 20                    // 用户定义的相关数据类型
BEGIN
        start：= TRUE；  // 赋初值
        setp.：= 10；
END_DATA_BLOCK
```

注意：

所用的 UDT 必须在源文件中数据块的前面。

带有相关功能块的数据块：

```
DATA_BLOCK DB30
TITLE = DB （FB） Example
FB30                          // 相关功能块
BEGIN
        start：= TRUE；    // 赋初值
        setp：= 10；
END_DATA_BLOCK
```

注意：

相关功能块必须在源文件中的数据块之前。

11.7.6 STL 源文件中用户定义数据类型的举例

```
Type  UDT20
STRUCT
     Start：BOOL；        // Bool类型的变量
     Setp： INT；         // INT类型的变量
     数值：WORD；         // WORD类型的变量
END_STRUCT；
END_TYPE
```

12 显示参考数据

12.1 可用参考数据概述

12.1.1 可用参考数据概述

生成并评估参考数据可使你的用户程序的调试和修改更加容易。参考数据作可作以下应用：

- 作为你的整个用户程序的一个概述
- 作为修改和测试的基础
- 完善你的程序文档

下表说明在各个视窗中可以摘取哪些信息：

视窗	目的
交叉参考列表	在存储区域 I、Q、M、P、T、C 以及 DB、FB、FC、SFB、SFC 块中由用户程序使用的地址概述。 用菜单命令 View>Cross Reference for Address，可以显示包括对所选地址的重复访问在内的所有交叉参考数据。
输入、输出及位存储（I、Q、M）赋值表。 定时器和计数器（T/C）赋值表	用户程序已占用的定时器和计数器（T 和 C）以及 I、Q、M 存储区中的位地址的概述；为故障诊断或修改用户程序奠定了重要基础
程序结构	在一个用户程序内块的分层调用结构，以及所使用的块及其嵌套层次的概述
未用符号	所有已在符号表中定义但未在用户程序的任何一个部分使用的符号概述。这些用户程序有可供使用的参考数据
无符号的地址	在有可供使用的参考数据的用户程序中，使用了但未在符号表中定义符号的绝对地址概述

所选用户程序的参考数据包括表中所列各项。可以为一个用户程序或多个用户程序生成并显示一种或多种列表。

同时显示多个视窗

在附加窗口显示其它列表可以让你：例如，

- 比较不同的S7用户程序的同一种列表。
- 显示某个列表的不同视窗，例如，不同显示的交叉参考列表在屏幕上并行排列。比如，你还可以在一个交叉参考列表中只显示S7用户程序的输入而在另一个列表中只显示输出。
- 同时打开一个S7用户程序的多个列表，例如，程序结构和交叉参考列表。

12.1.2 交叉参考列表

交叉参考列表给出了 S7 用户程序所用地址的概述。

一旦显示交叉参考列表就可以得到输入（I）、输出（Q）、位存储（M），定时器（T）、计数器（C）、功能块（FB）、功能（FC）、系统功能块（SFB）、系统功能（SFC）、I/O（P）和数据块（DB）这些存储区域中被 S7 用户程序使用的地地列表，在该列表中显示它们的地址（绝对地址或符号地址）及使用情况。显示在一个激活的窗口中。这个工作窗口的标题栏显示该交叉参考列表所属的用户程序名。

窗口中的每一行都对应着交叉参考列表的一个输入项。查寻功能让你很容易地找到指定的地址和符号。

当你显示参考数据时，交叉参考列表为缺省视窗。你可以改变这个缺省设置。

结构

一个交叉参考列表的输入项包括以下各栏：

栏	内容／含义
Address	绝对地址
Symbol	符号地址名
Block	使用该地址的块
Type	对有关地址的访问是读（R）和／或写（W）
Language/Details	用于生成块的编程语言的信息

只有为交叉参考列表选择了相应的选项才会显示符号（Symbol）、块（Block）、类型（Type）和语言/明细数据（Language/Details）。语言和明细数据的信息显示在一栏中并且只能整栏地激活或取消。块信息则依据该块编写时所用的编程语而定。

可以使用鼠标按照需要在屏幕上显示的交叉参考列表中设置栏宽。

分类

交叉参考列表的缺省选项是按存储区域分类。如果用鼠标点中栏标题，则可以按缺省分类标准对这一栏的输入项进行分类。

交叉参考列表格式示例

地址	符号	块	类型	语言	详细数据
I1.0	Motor on	OB2	R	STL	Nw 2 Inst 33 /0
M1.2	MemoryBit	FC2	R	LAD	Nw 33
C2	Counter2	FB2		FBD	Nw2

12.1.3 程序结构

程序结构概述了在一个 S7 用户程序内块的分层调用结构。你还可以对所用的块、它们的从属关系以及它们对局域数据的需求有一个概括的了解。

在“ Generating Reference Data（生成参考数据）”窗口，用菜单命令 View>Filter 打开一个标签对话框。在 “ Program Structure（程序结构） ” 标签中，你可以设置你想要程序结构如何显示。

你可以选其中之一：

- 树形结构和
- 父子结构（表格形式）

你可以指定是否要所有块都被显示或者分层结构是否从一个指定的起始块开始。

程序结构的符号

符号 含义

正常调用的块（CALL FB10）

无条件调用的块（UC FB10）

条件调用的块（CC FB10）

数据块

循环

循环且条件调用

循环且无条件调用

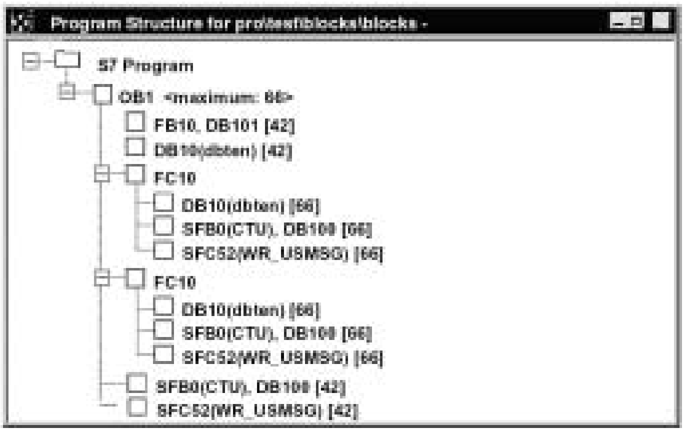
块未调用

- 在树形结构中识别并指示调用中的循环。
- 调用的分层结构中的循环被标记为不同的符号。
- 正常调用的块（CALL），条件调用的块（CC）或无条件调用的块（UC）被标记为不同的符号。
- 未被调用的块显示在树形结构的底部并且标记有黑叉。对于未被调用的块不会有更进一步的调用结构的分解。

树形结构

从某个特定的块开始的整个调用分层结构显示如下。

每个程序结构都只占有一个块作为根。这个块可以是 OB1 或任何其它的由用户预定为起始块的块。



如果要为所有的组织块（OB）生成程序结构，并且 OB1 不在 S7 用户程序中，或者指定的起始块不在程序中，你会被自动注意指定另一个块作为程序结构的根。

对于块的多重调用的显示，可以通过选项设置使之无效，这适用于树形结构，也适用于父子结构。

在树形结构中显示对局域数据的最大需求

为使你对显示的用户程序中的组织块的局域数据的需求有一个快速而概括的了解，在树形结构中可显示以下内容：

- 每个OB所要求的最大局域数据数以及
- 每个路经所要求的局域数据

你可以在 “ Program Structure ” 标签中激活或取消这个显示。

要显示所选块的局域数据要求，点击鼠标右键并选择上下文相关菜单中 “ Block Information（块信息） ” 菜单命令。

如果出现同步错误 OB（OB121，OB122），则在所要求的最大局域数据的数字后面显示一个加号以及该同步错误 OB 所另外要求的局域的数据数。

父子形结构

显示调用和被调用块。这里给出在 S7 用户程序中，每个块的调用关系对。

显示删除块

与册除块有关的行将显示为红色，并且在块后会显示字符串 “ ????? ” 。

12.1.4 输入、输出和位存储（I/Q/M）赋值表

赋值表将向你显示在用户程序中已经赋值的地址。这个显示是用户程序的故障查寻和修改的重要基础。

I/Q/M 赋值表的显示使你能概括地了解输入（I），输出（Q）和位存储存储区中哪个字节的哪一位被使用了。I/Q/M 赋值表在一个工作窗口中显示。这个工作窗口的标题栏显示该赋值表所属的 S7 用户程序名。

显示中每一行包含存储区的一个字节，在该字节中八个位，按其访问的情况标有代码。它还指示这个访问是否为一个字节、一个字或一个双字。

I/Q/M赋值表中的代码

·	地址未被访问，因而也没有赋值。
O	直接访问的地址。
X	间接访问的地址（字节、字或双字访问）。

I/Q/M赋值表的列

列	内容／含义
7 6 5 4 3 2 1 0	相应字节的位号
B	该字节以单字节访问的形式被占用
W	该字节以单字访问的形式被占用
D	该字节以双字访问的形式被占用

赋值表（I/Q/M）格式举例

下列例子显示了输入、输出和位存储（I/Q/M）的典型格式。

地址	7	6	5	4	3	2	1	0		B	W	D
QB0	O	X	X	O	X	X	X	X		O	·	·
QB1	·	O	·	·	O	·	O	·		·	·	·
IB0	O	O	O	·	O	·	O	·		·	·	·
IB1	·	·	·	·	·	·	·	·		·	·	·
MB0	X	X	X	X	X	X	X	X		·	O	·
MB1	X	X	X	X	X	X	O	X		·	·	·

第一行给出了输出字节 QB0 的分配。地址 QB0 是以字节方式来访问的。同时，输出位 Q0.4 和 Q0.7 还用作位访问。因此，在列 4 和列 7 中有一个“ O ”。在列“ O ”，“ 1 ”，“ 2 ”，“ 3 ”，“ 5 ”和“ 6 ”中有一个“ X ”表示字节访问。在列 B 中有一个“ O ”出现，是因为它是一个地址 QB0 的字节访问。

12.1.5 定时器和计数器（T/C）赋值表

赋值表将向你显示在用户程序中已经赋值的地址。这个显示是用户程序的故障查寻和修改的重要基础。

T/C 赋值关系表向你显示计数器和定时器应用的一个概述。

T/C 赋值关系表在一个工作窗口中显示。这个工作窗口的标题栏显示该赋值表所属的 S7 用户程序名。每行包含 10 个计数器或定时器。

T/C赋值关系表中的代码

·	未使用
x	使用

赋值关系列表（T/C）格式举例

	0	1	2	3	4	5	6	7	8	9
T 00-09	·	X	·	·	·		X	·	·	·
T 10-19	·	·	X	·	·	·	·	X	·	X
T 20-29	·	·	·	·	X	·	·	·	·	·
C 00-09	·	·	X	·	·	·	·	X	·	·
C 10-19	·	·	·	·	·	·	·	·	·	X
C 20-29	·	·	·	·	·	·	·	·	·	·
C 30-39	·	·	·	·	X	·	·	·	·	·

在此例中，定时器 T1，T6，T12，T17，T19，T24 和计数器 C2，C7，C19，C34 是已经赋值的。

这些列表是按字母顺序分类的。你可以通过点中栏标题来对条目分类。

12.1.6 未使用的符号

你可以看到具有下列特征的所有符号的一个概述显示：

- 已在符号表中定义的符号。
- 有参考数据的存在却未在用户程序的任何一部分中使用的符号。

它们在一个活动窗口中显示。这个工作窗口的标题栏显示该列表所属的用户程序名。

窗口中显示的每行对应一个列表输入项。由地址，符号、数据类型和注释组成一行。

列	内容／含义
Symbol	符号名称
Address	绝对地址
Date-Type	地址的数据类型
Comment	符号表中的地址的注释

未用符号的格式举例

Symbol	Address	Data Type	Comment
MCB1	I 103.6	BOOL	Motor circuit breaker 1
MCB2	I 120.5	BOOL	Motor circuit breaker 2
MCB3	I 121.3	BOOL	Motor circuit breaker 3

你可以通过点中栏标题来对条目分类。

12.1.7 没有符号的地址

当你显示没有符号的地址列表时，可以得到在 S7 用户程序中使用了但未在符号表中定义的地址的列表。它们在一个活动窗口中显示。这个工作窗口的标题栏显示该列表所属的用户程序名。

每一行包括地址和该地址在用户程序中使用的次数。

例如：

地址	次数
Q 2.5	4
I 23.6	3
M 34.1	20

该列表按照地址来存储。

12.1.8 显示 LAD、FBD 和 STL 的块信息

在交叉参考列表和程序结构中可以显示梯形图、功能块图和语句表的块信息。这个信息包括块的语言和明细数据。

在“ Program Structure（程序结构）”视窗中，用菜单命令 View>Block Information 或通过鼠标右键可显示块信息。显示依据所选的表达方式而定，“ Program structure ” 标签中，在筛选设定下可以选择 “ 父子结构 ” 或 “ 树形结构 ” 两种表达方式。

在 “ Cross References（交叉参考） ” 视窗中，使用命令 View>Filter，可切换显示或不显示块信息。

- 激活 “ Filter（筛选器） ” 对话框的 “ cross Reference（交叉参考） ” 标签中的 “ Block language and details（块语言和明细数据） ” 复选框以显示块信息。

语言	段	语句	指令
STL	Nw	Inst	/
LAD	Nw		
FBD	Nw		

块信息根据块编写时的编程语言的不同而变化，用缩写显示。

Nw 和 Inst 指示在哪个段和哪条语句中使用了该地址（交叉参考列表）或调用了该块（程序结构）。

为可选编程语言显示块信息

如果安装了相应的可选软件包则可访问有关块信息的在线帮助。

12.2 用参考数据

12.2.1 参考数据显示方式

以下是可用来显示参考数据的方法：

从SIMATIC管理器中显示

1. 在离线组件视窗的项目窗口中，选择“ Blocks（块） ”文件夹。
2. 选择菜单命令Options>Reference Data>Display。

从编辑器窗口显示

1. 在“ Blocks ”文件夹中打开一个块。
2. 在编程语言编辑器的窗口中，选择菜单命Options>Reference Data。

将显示“ Customize（用户自定义） ”对话框。在此，用户可以首先选择所显示的窗口。缺省窗口为应用程序中显示参考数据最后关闭的窗口。用户可以取消对话，以便以后调用。

从编译的块直接显示

可以从语言编辑器中的一个编译的块直接显示参考数据，以获得你的用户程序的当前概况。

12.2.2 在另外的工作窗口显示列表

使用菜单命令 Window>New Window，可以打开另一个工作窗口，显示参数据的其它视窗（例如，未使用的符号列表）。

使用菜单命令 Reference Data>Open，可以为以前没有显示的参考数据打开一个工作窗口。选择“ View（视窗） ”菜单中的某个命令或工具栏中相应的按钮，可以改变为其它的参考数据的视窗：

参考数据视窗	显示该参考数据视窗的菜单命令
没有符号的地址	View > Addresses Without Symbols
未使用的符号	View > Unused Symbols
I/Q/M 赋值关系列表	View > Assignment > Inputs， Outputs， and Bit Memory
T/C 赋值关系列表	View > Assignment > Timers and Counters
程序结构	View > Program Structure
交叉参考列表	View > Cross References

12.2.3 生成并显示参考数据

生成参考数据：

1. 在 SIMATIC 管理器中，选择你要为之生成参考数据的块文件夹。
2. 在 SIMATIC 管理器中选择菜单命令 Options>Reference Data>Generate。

在生成参考数据之前，计算机检查是否已有参考数据，如果有，是否为当前数据。

- 如果有参考数据，则已被生成。
- 如果已有参考数据不是当前的，可以选择是否刷新这些参考数据或是否重新全部生成。

显示参考数据：

使用菜单命令 Options>Reference Data>Display，可以显示参考数据。

在显示参考数据前，要作一个检查以确定是否已有参考数据存在，以及存在的参考数据是否是当前的。

- 如果没有参考数据存在则生成它们。
- 如果有不完整的参考数据存在，显示的对话框会注意该参考数据不一致。然后你可以决定是否要刷新该参考数据并且到什么程度。有以下几种可能：

选择	含义
只为修改过的块	参考数据只为那些修改过的或新生成的块而刷新；任何已被删除的块的信息会从参考数据库中去掉
为所有的块	从临时存储区中为所有的块重新生成参考数据
不刷新	参考数据不被刷新

为刷新参考数据，块被再次编译。调用适当的编译器以编译每一个块。使用菜单命令 View>Update，可以刷新已显示在活动窗口中的参考数据的视窗。

12.2.4 在程序中快速查找地址的位置

编程时使用参考数据，可将光标定位于某一个地址在程序中的不同位置上。要这样做，你必须有最新的参考数据。但是，你不必启动应用程序来显示参考数据。

基本步骤

1. 在 SIMATIC 管理器中，选择菜单命令 Options>Reference Data>Generate，生成当前的参考数据。只有当没有参考数据时或参考数据是旧的，这一步才有必要。
2. 在一个打开的块中选择地址。
3. 选择菜单命令 Edit>Go To>Location。
   现在显示一个对话框，包含该地址在程序中的位置列表。
4. 如果你还要显示与被调用的地址的物理地址或地址区域相重叠的那些地址的位置，选择选项“ Overlapping access to memory areas（地址区域的重叠访问）”。则这种“ 地址 ”栏被加到表中。
5. 在列表中选中某位置并点击“ Go To ”按钮。

如果打开对话框时参考数据不是最新的，则会有与之相关的信息出现。然后就可以刷新参考数据了。

位置列表

对话框中的位置列表包含以下明细数据。

- 使用了该地址的块。
- 块的符号名，如果该块有符号名的话。
- 详细数据，例如，位置信息，以及依据块或源文件（SCL）最初所用的编程语言而定，如果合适的话，还会有指令显示。
- 依语言而定的信息。
- 对该地址的访问类型：只读（R）、只写（W）、读且写（RW）、未知（?）。
- 块语言。

可以对位置的显示作筛选，比如，只显示对某一个地址的写访问。关于此对话框，在线帮助为你提供了更多的关于在该区域中输入什么以及显示的其它信息的详细信息。

注意

参考数据只能离线存在。因此该功能总是使用离线块的交叉参考。即使在一个在线块中调用该功能也是这样。

12.2.5 使用地址位置表的示例

要判定输出 Q1.0（直接/间接）在哪个位置被置位了。用以下的 STL 指令的 OB1 作为一个示例：

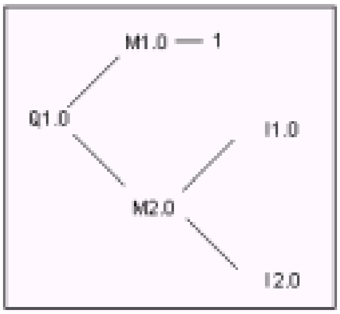
```
Network 1：  ...........
A Q1.0          //  在本例中
= Q1.1          //  无关

Network 2：
A M1.0
A M2.0
=Q 1.0          //  赋值

Network 3：
//  注释行
SET
=M1.0           //  赋值

Network 4：
A   I   1.0
A   I   2.0
=M2.0           //  赋值
```

得到以下的 Q1.0 的赋值关系树：



然后按如下进行：

1. 在 LAD/STL/FBD 编辑器中将光标位于 OB1 的 Q1.0（NW1，Inst1）上。
2. 选择菜单命令 Edit>Go To>Location 或用鼠标右键选择“ Go to Location ”。现在对话框中显示 Q1.0 的所有赋值关系：

OB1	Cycle Execution	NW 1	Inst 3	/=	W	STL
OB1	Cycle Execution	NW 2	Inst 4	/A	R	STL

3. 在对话框中用“ GO TO ”按钮，跳到编辑器中的“ NW2 Inst3（第二段第三条指令）”：

```
Network 2：
A M1.0
A M2.0
= Q 1.0
```

4. 现在必须检查对 M1.0 和 M2.0 的赋值。首先将光标位于 LAD/STL/FBD/编辑器中的 M1.0 上。
5. 选择菜单命令 Edit>Go To>Location 或用鼠标右键选择“ Go to Location ”。现在对话框中显示 M1.0 的所有赋值关系：

B1	Cycle Execution	NW 3	Inst 2	/=	W	STL
OB1	Cycle Execution	NW 2	Inst 1	/A	R	STL

6. 用对话框中的“ Go To ”按钮跳到编辑器中的“ NW3 Inst2（第三段第二条指令）”。
7. 在 LAD/STL/FBD 编辑器中的第三段中可以看到，对 M1.0 的赋值不重要（因为总是 1），因此应该检查 M2.0 的赋值。

   在早于V5的STEP 7版本中，这时你就得重新把整个赋值关系链完整地查一遍。按钮“ >> ”及“ << ”能使这个操作简单一些：

8. 将打开的对话框“ Go to Location ”拖至前台，或从 LAD/STL/FBD 编辑器中你的当前位置上调用功能“ Go to Location ”。

9. 点击“ << ”按钮一次或两次直至显示 Q1.0 的位置；选择最后一个跳转位置“ NW2 Inst3 ”。

10. 用“ GO TO ”按钮（如第三点中所示）从地址位置对话框跳到编辑器中的“ NWI Inst3 ”。

```
Network 2：
A M1.0
A M2.0
= Q 1.0
```

11. 在第 4 点中，检查了 M1.0 的赋值。现在要检查所有（直接/间接）对 M2.0 的赋值。将光标放在编辑器中 M2.0 上并调用功能“ Go to Location：”所有对 M2.0 的赋值都显示出来：

OB1	Cycle Execution	NW 4	Inst 3	/=	W	STL
OB1	Cycle Execution	NW 2	Inst 2	/A	R	STL

12. 用“ Go To ”按钮跳到 LAD/STL/FBD 编辑器中的“ NW4 Inst3 ”：

```
Network 4：
A  I  1.0
A  I  2.0
= M2.0
```

13. 现在要检查对 I1.0 和 I2.0 的赋值。在本例中不再描述这一过程，因为可以按照和以前一样的方式进行（从第 4 点开始）。

通过在 LAD/STL/FBD 编辑器和地址位置对话框之间进行切换，可以在你的用户程序中找到并检查相关的位置。

13 检查块的一致性和作为块特性的时间标记

13.1 检查块的一致性

简介

如果必须使用或扩展单个块的接口或代码，会导致时间标记冲突。同样，时间标记冲突会造成调用块和被调块或基准块之间的不一致，从而带来繁重的修正工作。

“检查块的一致性”功能可以避免这种修正工作。“检查块的一致性”功能可以清除所有时间标记冲突和不一致块的一大部分。对于不一致性不能自动消除的块，该功能可以将你切换到相应的编辑器中，在此，可以进行修改。所有块的不一致性都可以消除，并一步一步地进行编译。

要求

只有 STEP 7 V5.0 Service Pack 3 生成的项目，才能检查块的不一致性。对于较早的项目，在进行块的一致性检查时，你必须首先编译每一项（菜单命令 Program > Compile All）。

对于使用任选软件包生成的块，必须安装任选软件包，才能检查块的一致性。

启动块的一致性检查

在开始进行块的一致性检查时，应检查块接口的时间标记，并且会造成块的不一致性的块将高亮显示在树视窗中（Dependency Tree/Reference Tree（相关树/参考树））。

1. 选择菜单命令 Program > Compile。

   STEP 7 可以自动识别相关块的编程语言，并调用相应的编辑器。应尽可能地自动修正时间标记冲突和块的不一致性，并编译块。如果不能自动清除块中的时间标记冲突或不一致性，在输出窗口中将出现错误信息（参见第 2 步）。对于树视窗中的所有块，将自动重复这一过程。

2. 如果在编译运行时，不能自动清除所有块的不一致性，在输出窗口中，相应的块将标记为错误信息。将鼠标放在相应的错误处，并使用鼠标右键，调用弹出菜单中的错误显示。打开相关错误，程序将跳到被修改的位置。清除所有块的不一致性，保存并关闭块。对于所有标记为错误的块，重复这一过程。
3. 重新执行第 1 步和第 2 步。重复该过程，直至在信息窗口中不再有错误显示。

“ 块的一致性检查 ” 视窗

“ 块的一致性检查 ” 输出窗口

13.2 作为块特性的时间标记及时间标记冲突

块中包含一个代码时间标记和一个接口时间标记。这些时间标记可以被显示在块特性的对话框中。你可以监控 STEP 7 程序使用时间标记的一致性。

STEP 7 在进行时间标记比较时，如发现违背了规则，就会显示时间标记冲突。以下是可能出现的违背规则的几种情形：

- 一个被调用的块比调用它的块的时间标记新（CALL）。
- 一个被参考的块比使用它的块的时间标记更新。

第二种违背规则的示例：

- 一个UDT比使用它的块的时间标记更新；这些块可以是一个DB或其它的UDT，或者在变量声明表中使用了该UDT的FC、FB、OB。
- 一个FB比其相应的背景数据块的时间标记更新。
- 一个FB2在FB1中被定义为多重背景，并且FB2的时间标记比FB1的更新。

注意

既使接口时间标记之间的关系是正确的，不一致性仍可能出现：

- 被参考的块的接口的定义与在它被使用的区域中的定义不匹配。

这些不一致就是所说的接口冲突。它们可能会出现在，比如，当块由不同的程序中拷贝出来或者当一个 ASCII 源文件编译时，不是程序中所有的块都生成了。

13.3 逻辑块中的时间标记

代码时间标记

块生成时的时间和日期就存储在这里。该时间标记会被刷新：

- 当程序代码被修改
- 当接口描述被修改
- 当注释被修改
- 当ASCII源文件第一次生成并编译
- 当块特性（“ properties ” 对话框）被修改

接口时间标记

该时间标记会被刷新：

- 当接口描述改变（修改了数据类型或初始值、增加新参数）
- 如果接口结构上有改变，当ASCII源文件第一次生成并编译。

该时间标记会被刷新：

- 当符号被修改
- 当变量声明表中的注释被修改
- 当TEMP（临时）区域有改变

块调用的规则

- 被调用的块的接口时间标记必须比调用块的代码时间标记旧。
- 只有当设有任何一个调用这个块的块被打开时，才可以修改这个块的接口。否则，当你修改了这个块之后再存储调用它的块，就无法从时间标记上识别这种不一致性。

时间标记冲突出现的程序

当调用块被打开时，时间标记冲突就显示出来。在对 FC 或 FB 接口作修改之后，在调用块中对这个块的所有调用以扩展的形式显示。

如果一个块的接口修改了，则所有调用该块的块都必须相应修改。

在对一个 FB 的接口修改后，已有的多重背景的定义及背景数据块必须刷新。

13.4 共享数据块的时间标记

代码时间标记

该时间标记会被刷新：

- 当一个ASCII源文件首次生成时
- 当一个ASCII源文件编译时
- 当在块的声明表显示方式下或在数据显示方式下作出修改时

接口时间标记

该时间标记会被刷新：

- 当在声明表显示方式下，接口描述被修改（修改数据类型或初始值、有新参数）时。

13.5 背景数据块的时间标记

一个背景数据块用于为功能块存储形式参数和静态数据。

代码时间标记

背景数据块生成的时间和日期存储在这里。当你在背景数据块的数据显示方式下送入实际值时，该时间标记被刷新。由于背景数据块的结构来自于相关的功能块（FB）或系统功能块（SFB），所以用户无法改变其结构。

接口时间标记

当一个背景数据块生成时，存储与其相关的 FB 或 SFB 的接口时间标记。

无冲突打开的规则

FB/SFB 及其相关的背景数据块的接口时间标记必须相匹配。

时间标记冲突出现的程序

如果你修改了 FB 的接口，该 FB 的接口时间标记则被刷新。当你打开一个与其相关的背景数据块时，则显示时间标记冲突的报告。因为此时背景数据块的时间标记与 FB 的不再匹配。在数据块的声明部分，接口显示由编译器生成的符号（伪符号）。该背景数据块此时只能看不能改。

为改正这种类型的时间标记冲突，你必须为修改过的 FB 重新生成背景数据块。

13.6 UDT 的以及源自于 UDT 的数据块的时间标记

用户定义的数据类型（UDT）可以用来生成大量的、具有相同结构的数据块。

代码时间标记

任何一种修改都会使代码时间标记刷新。

接口时间标记

当接口描述改变（修改数据类型或初始值、增加新参数），接口时间标记会被刷新。

当 ASCII 源文件编译时，UDT 的接口时间标记也会被刷新。

无冲突打开的规则

- 用户定义的数据类型的接口时间标记必须比使用该数据类型的逻辑块的接口时间标记旧。
- 用户定义的数据类型的接口时间标记必须与源自于该UDT的数据块的时间标记一致。
- 用户定义的数据类型的接口时间标记必须比二级UDT的时间标记新。

时间标记冲突出现的程序

如果你修改了一个 UDT 的定义，而该 UDT 被用于某个数据块、功能、功能块或另一个 UDT 的定义，当使用这个 UDT 的块被打开时，“ STEP 7 将报告时间标记冲突。

UDT 的元素被显示为一个展开结构。所有变量名被系统预置值覆盖。

14 组态报文

14.1 报文概念

14.1.1 报文概念

报文（Message）功能允许你在可编程控制器运行过程中快速地检测、定位及纠正错误，从而可显著地减少工厂的停工期。

要想有报文输出，首先要进行组态。

使用 STEP 7，你可以生成并编辑报文，这些报文与已被赋予报文文本和报文属性的事件相连。还可以编译这些报文并将它们显示在显示装置上。

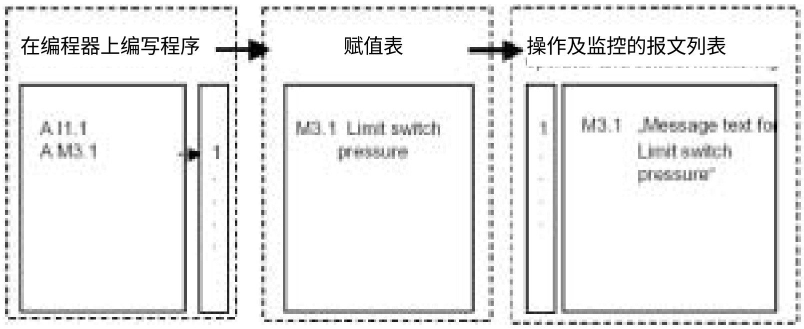
14.1.2 什么是不同报文方法?

有许多生成报文的不同方法。

位报文

位报文需用编程器完成以下三步：

- 在编程设备上生成用户程序并设置所需要的位。
- 使用任意文本编辑器生成一张赋值表，在该表中一条报文文本被赋给一个报文位（如，M3.1=limit switch pressure）。
- 在赋值表的基础上，在操作面板上生成报文文本列表。

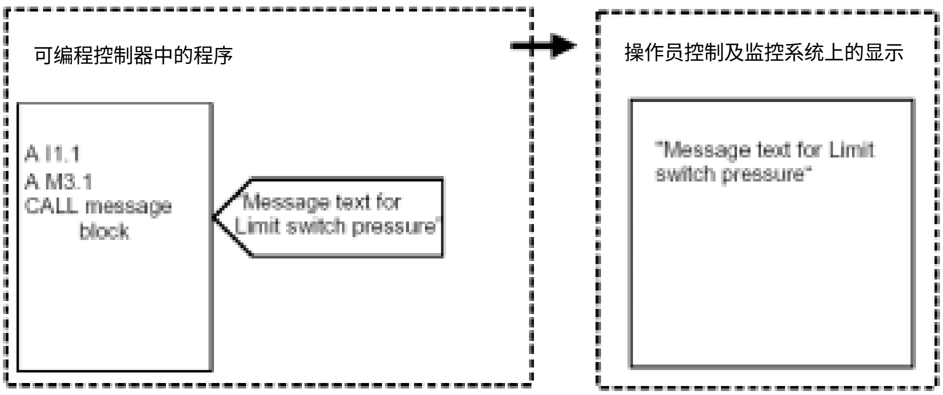


操作员接口系统不断地循环查询可编程控制器，查看是否有报文位发生变化。如果可编程控制器有信号变化，相应的报文则会显示。报文从操作员接口系统接收时间标记。

报文编码

报文计数功能只需要编程器完成一个步骤：

- 在编程器上生成用户程序，设置所需要的位，并在编程时直接将所需的报文文本赋值给这个位。



不需要 PLC 的循环查询。当 PLC 的信号改变时，相应的报文号传送到操作员界面系统，并显示相应的报文。报文从 PLC 上按收到时间标记，并可以更准确地跟踪。

14.1.3 选择一种报文方法

概述

下表是不同报文方法的特性及要求：

报文编码	位报文
• 在编程设备和操作面板的公共数据库中进行报文处理 • 总线负载低（可编程控制器信号激活） • 报文从可编程控制器接收时间标记	• 编程设备和操作面板没有公共数据库 • 总线负载高（操作面板查询） • 报文从操作面板接收时间标记

报文编码方式可以识别以下三种类型的报文：

STEP 7 支持更加用户友好的报文编码方式，我们将在下面进行详细的描述。

与块相关的报文	与符号相关的报文	用户定义的诊断报文
• 与程序同步 • 使用 WinCC 和 ProTool 显示（只有 ALARM_S） • 可用于 S7-300/400 • 用报文块编程： - ALARM - ALARM_8 - ALARM_8P - NOTIFY - ALARM_S（Q） - AR_SEND • 传送到操作面板 • 通过 PLC-OS 连接组态用于 WinCC • 通过 ProTool 功能用于 ProTool	• 与程序异步 • 用 WinCC 显示 • 只用于 S7-400 • 用符号表组态 • 通过系统数据块（SDB）传送到可编程控制器 • 通过 PLC-OS 连接组态传送至操作面板	• 与程序同步 • 在编程设备的诊断缓冲区中显示 • 可用于 S7-300/400 • 使用报文块（系统功能）编程 - WR_USMSG • 不传送到操作面板

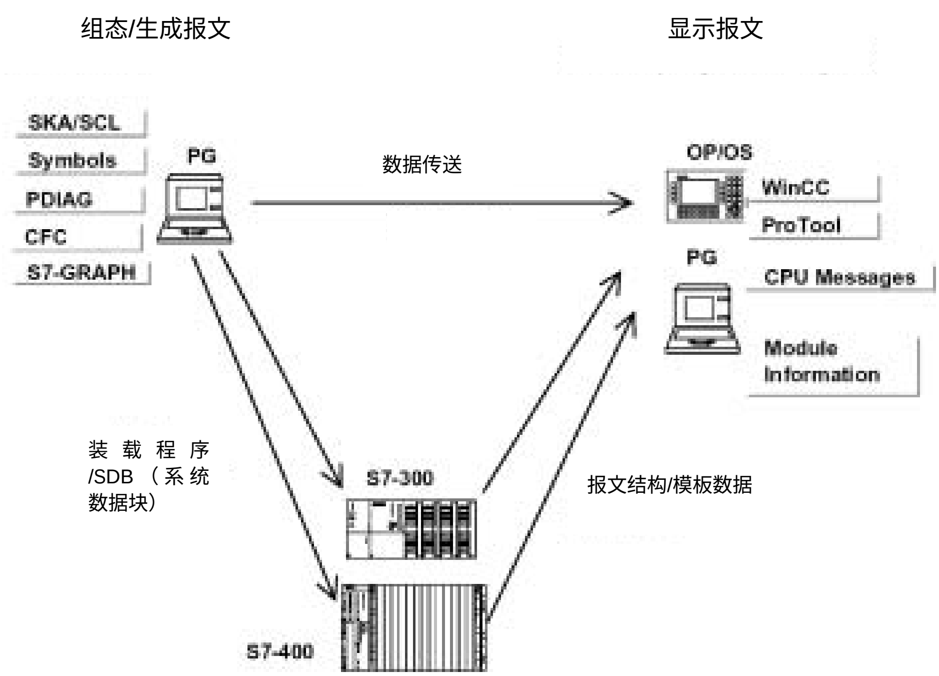
报文编码示例

报文方式	应用程序
与块相关的报文	用于报告与程序同步的事件，比如，显示控制器到达了一个定限值
与符号相关的报文	用于报告与程序无关的事件，比如，一个开头装置被监控
用户定义的报文	用于对每个 SFC 的调用报告诊断缓冲区中的诊断事件

14.1.4 SIMATIC 组件

概述

下图所示为与组态和显示报文有关的 SIMATIC 组件的概观。下表中列出了一条报文可能的组成部件：



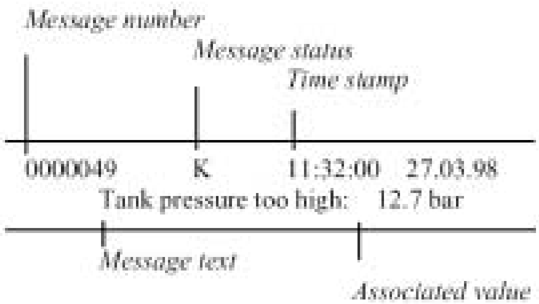
14.1.5 报文的部件

一条报文如何显示要看报文方式、使用的报文块以及显示设备。

部件	描述
时间标记	当报文事件出现时，在可编程控制器中生成
报文状态	可能有以下状态：到来，离开，离开无应答，离开有应答
相关值	有时报文可被赋予一个过程值，该值可被所用的报文块进行评估
图像	如果系统故障，出现的报文可随后显示在操作站上
报文编号	由系统分配一个在整个项目中唯一的号码用以识别该报文
报文文本	用户组态

举例

下列所示，是操作面板上的一个报警信息。



14.1.6 指定信息号码

报文通过其编码来识别，该编码在整个项目中唯一。要做到这一点，在总的数值范围（1 到 2097151）内给每个 STEP 7 程序分配一个数码范围。如果你作程序拷贝并产生了冲突，即；在目标范围内同一范围的编码已被赋值——则新程序必须被分配一个新的数码范围。如果这种情况出现，STEP 7 会自动打开一个对话框，你可以在该对话框中指定新的数码范围。

如果还没有组态报文，你还可以使用菜单命令 Edit>Special Object Properties>Message Numbers。设置或改变 S7 程序的数码范围。

报文数码范围缺省设置为 20000 步。

14.2 设定和编辑与软件块相关的信息

14.2.1 设定和编辑与软件块相关的信息

与块相关的报文被赋值给一个块（FB）。要生成一个块相关的报文，可使用系统功能块（SFB）和系统功能（SFC）作为报文块。

14.2.2 有哪些报文块可用?

你可以在下列报文块中选择，每个块包含一个已编程的报文功能：

- SFB33 “ ALARM ”
- SFB34 “ ALARM_8 ”
- SFB35 “ ALARM_8P ”
- SFB36 “ NOTIFY ”
- SFC18 “ ALARM_S ”和SFC17 “ ALARM_SQ ”
- SFB37 “ AR_SEND ”（送至存档）

参考块的在线帮助，你会得到更详细的信息。

什么时候用什么样的报文块?

下表可以帮助你决定为特定任务选择哪个报文块。可根据以下情况选择报文块：

- 块中可用的通道以及因此每个块要监控的信号
- 报文是否要应答
- 指定相关值的选项
- 使用的显示设备
- 所用CPU 的技术规格

报文块	通道	应答	相关值	WinCC 显示	ProTool 显示	CPU 信息 /S7 状态显示	PLC	解释
ALARM SFB33	1	可能	最多 10 个	有	无	无	S7-400	每个上升沿或下降沿到来时发送报文
ALARM_8 SFB34	8	可能	无	有	无	无	S7-400	一个或多个信号的每个到来或离开边沿发送报文
ALARM_8P SFB35	8	可能	最多 10 个	有	无	无	S7-400	同 ALARM_8
NOTIFY SFB36	1	无	最多 10 个	有	无	无	S7-400	同 ALARM
AR_SEND SFB37	1			有	无	无	S7-400	用于发送文档
ALARM_SQ SFC17	1	可能	1	有	有＊	有	S7-300/ S7-400	不是边沿变化而是每个 SFC 调用产生报文
ALARM_S SFC18	1	无	1	有	有＊	有	S7-300/ S7-400	不是边沿变化而是每个 SFC 调用产生报文
＊取决于 OP 类型								

14.2.3 形式参数、系统属性以及报文块

作为报文编码输入的形式参数

在你的用户程序中每个报文或每组报文需要一个形式参数，该形参在程序的变量声明表中定义为输入变量，从而可以使用这个形参作为报文编码输入，形成报文的基础。

如何用系统属性提供形参

作为开始组态报文的前提要求，你必须首先按如下所述，用系统属性提供形式参数：

1. 为参数“ S7_server ”和“ S7_a_type ”增加下述的系列属性：“ S7_server ”和“ S7_a_type ”
2. 根据你在程序指令中调用的报文块，给系统属性赋值。赋给“ S7_server ”的总是“ alarm_archiv ”，而赋给“ S7_a_type ”的值要依据被调用的块而定。

系统属性和相应的报文块

报文块本身不会在报文管理器中作为一个对象显示出来；取而代之，显示中包含系统属性“ S7_a_type ”的值。这些值作为报文块具有相同的名字，以 SFB 或 SFC 的形式存在（例外：“ alarm_s ”）。

S7_a_type	报文块	描述	特性
alarm_8	ALARM_8	SFB34	8 通道，能被应答，无相关值
alarm_8p	ALARM_8P	SFB35	8 通道，能被应答，每个通道的相关值最多 10 个
notify	NOTIFY	SFB36	1 通道，不能被应答，相关值最多 10 个
alarm	ALARM	SFB33	1 通道，能被应答，相关值最多 10 个
alarm_s	ALARM_S	SFC18	1 通道，不能被应答，最多 1 个相关值
alarm_s	ALARM_SQ	SFC17	1 通道，能被应答，最多 1 个相关值
ar_send	AR_SEND	SFB37	用于发送文档

参考块的在线帮助，你会得到更详细的信息。

如果你在你用户程序中所使用的报文块为具有相应系统属性的 SFB 或 FB，并作为多重背景调用，系统属性将自动赋值。

14.2.4 报文模板及报文

报文组态允许你按不同的步骤去生成报文模板或报文。至于采取何种步骤，要依据你通过哪个报文类型块去访问报文组态而定。

报文类型块即可以是一个功能块（FB）也可以是一个背景数据块

- 使用FB可以生成一个报文模板作为生成报文的样板。你为报文模板所作的所有项都自动地输入到报文中。如果你将一个背景数据块赋给一个功能块，则该背景数据块的报文按照报文模板和所赋值的报文编码自动生成。
- 对背景数据块来说，你可以修改报文，该报文是基于特定背景数据的报文模板而生成的。

这里明显的差别是：报文编码是赋给报文的而不是给报文模板的。

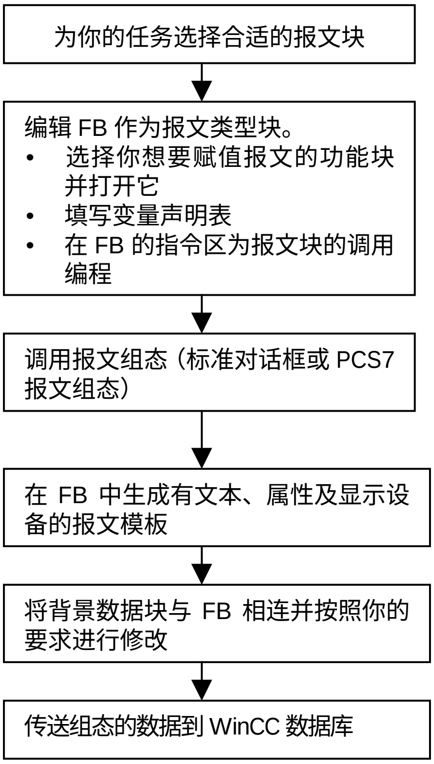
为报文模板进行数据加锁

报文组态允许你为依事件而定的报文输入文本和属性。比如，你可以指定在特定的显示设备上怎样显示报文。为使报文生成更容易，可以使用报文模板。

- 当你为报文模板输入数据（属性和文本）时，可以指定它们是否被加锁。若带有加锁属性则输入框旁边会增加一个键符。已加锁的文本在“ Locked ”栏中显示一个复选标志。
- 使用报文模板的“ 加锁数据 ”，你不能在特定的背景数据报文中进行修改。数据只能被显示。
- 如果确实需要修改，必须回到报文模板，去掉锁，并在这里修改。这些修改不会自动加到以前生成的背景数据中。

14.2.5 生成与块相关的报文

基本程序



编程报文类型块（FB）

1. 在 SIMATIC 管理器中选择一个你想为它生成块相关报文的功能块（FB），双击打开它。

   结果：打开被选中的块并显示在“ LAD/STL/FBD ”窗口。

2. 填写变量声明表。因为对每一个被调用的报文块，必须在调用功能块中声明变量。

   在变量声明表中，在“ Declaration ”栏中输入下列变量：

   - 在声明类型“ in ”中输入一个符号名作为报文块的输入，例如，“ Mess01 ”（为报文输入 01）以及类型（必须是“ DWORD ”，没有初始值）。
   - 在声明类型“ stat ”中为调用的报文块输入符号名，例如，“ alarm ”及相应的类型，这里为“ SFB33 ”。

3. 在功能块的指令部分，插入对选中的报文块的调用，这里为“ CALL alarm ”，用RETURN键完成输入。

   结果：被调用的报文块（这里是 SFB33）的输入变量显示在功能块的指令区。

4. 将第2步中为报文块输入分配的符号名，这里是“ Mess01 ”赋给变量“ EV_ID ”并确认系统属性被用于报文组态。

   结果：如果“ Name ”栏没有选中，则在该栏中会出现一个标志。选中的块则被设置为报文类型块。所要求的系统属性（如 S7_server 和 S7_a_type）及相应值会自动被赋值。

5. 在功能块中为所有的报文块的调用重复步骤2至4。
6. 用菜单命令File>Save保存该块。
7. 关闭“ LAD/STL/FBD ”窗口。

打开报文组态对话框

- 在SIMATIC管理器中选择菜单命令Edit>Special Object Properties > Message。

  结果：STEP 7 报文组态对话框（标准对话框）打开。在 PCS7 报文组态下可以找到有关打开 PCS7 报文组态功能的信息。

生成报文模板

1. 选择显示出来的报文块并且在“ Attributes ”和“ Text ”选项卡中输入所需的属性和报文文本。

   如果你选择的是一个多通道报文块（例如，“ ALARM_8 ”），你可以将其自己的报文文本赋值给每个报文子编码。属性适用于所有报文子编码。

2. 击“ New Device ”按钮将所需的显示设备赋给报文模板，并在“ Add Display Device ”对话框中选择所需的显示设备。

在下面的制表页中为显示设备输入所需文本及属性。用“ OK ”退出对话框。

注意

当编辑特定的显示设备文本和属性时，请阅读随同你的显示设备提供的文件。

生成背景数据块

1. 当你生成报文模板时，可以给它关联一个背景数据块，并且为这些数据块编辑特定背景报文。

   要实现这一步，需在 SIMATIC 管理器中打开一个块，在该块中要调用前面组态的功能块，例如，双击“ OB1 ”。在该 OB 块的指令部分输入调用指令（“ CALL ”），FB 的名字和号码以及作为背景与 FB 相关的背景数据块。用 RETURN，确认你的输入。

   例如：输入“ CALL FB1，DB1 ”。如果 DB1 不存在，对是否生成背景数据块的注意回答“ Yes ”。

   结果：背景数据块生成了。在 OB 的指令部分显示相关 FB 的输入变量，这里的示例是“ Mess01 ”，以及由系统分配的报文编码，这里为“ 1 ”。

2. 用菜单命令 File>Save，存储 OB，并关闭“ LAD/STL/FBD ”窗口。

编辑报文

1. 在 SIMATIC 管理器中，选择已生成的背景数据块，例如“ DB1 ”，选择菜单命令 Edit>Special Object Properties>Message，打开报文组态对话框。

   结果：“ Message Configuration ”对话框打开，显示所选中的带有由系统分配的报文编码的背景数据块。

2. 在适当的表栏中，输入相应的背景数据块所需的修改，如果需要，可增加其它的显示设备。用“ OK ”退出对话框。

   结果：为选中的背景数据块所作的报文组态就完成了。

传送组态数据

- 将组态的数据传送至WinCC数据库（通过PLC-OS连接组态）或ProTool数据库。

14.2.6 PCS 7 报文组态

为编辑报文模板以及将报文送出到 WinCC 显示设备，STEP 7 中的 PCS7 报文组态功能提供用户友好方式如下：

- 显示设备组态的简化（自动生成）
- 报文属性和文本输入的简化
- 确保报文标准化

打开PCS 7报文组态功能

1. 在 SIMATIC 管理器中，选择你想要编辑报文文本的块（FB 或 DB），使用菜单命令 Edit > Object Properties，打开输入系统属性的对话框。

2. 在显示的表格中输入以下系统属性：

   - 属性：“ S7_alarm_ui ”以及值“ 1 ”用于非事件驱动 PCS7 报文（数值“ 0 ”将关闭 PCS7 报文组态）。在 LAD/STL/FBDP 中可以设定特性参数。以后生成并赋值到相应 FB 的 DB，将采用这些属性，并且可以切换，与其报文类型无关（FB）。

   - 属性：“ S7_alarm ”和数值“ alarm_16 ”（如果所选块为事件驱动通讯块（EDC））。对于事件驱动通讯，不能赋值特性参数。赋值给 DB 的这些 FB 的报文文本在 SIMATIC Manager 中不能编辑。

注意

当你输入系统属性时，语法检查正在运行，不正确的输入被标定为红色。

3. 用“ OK ”退出对话框。
4. 选择菜单命令 Edit>Special Object Properties>Message。

   结果：“ PCS7 Message Configuration ”对话框被打开。

编辑报文模板

1. 在 SIMATIC 管理器中，选择你想编辑报文文本的 FB，打开 PCS7 报文组态对话框。

   结果：对每一个在 FB 中声明了变量的报文块，有一个选项卡显示在对话框中。对于事件驱动通讯块，在对话框中将出现两个选项卡。
2. 为报文的组件“ Origin ”、“ OS area ”及“ Batch ID ”填写文本框。
3. 输入报文级别并为报文块使用的所有事件输入事件文本，指定每个事件是否必须一一应答。
4. 对于用于所有背景的、不能被修改的报文组件，点击“ Locked ”复选框。

编辑报文

1. 在 SIMATIC 管理器中，选择你想编辑报文文本的背景数据块，打开 PCS7 报文组态对话框。
2. 不要修改未加锁的特定背景报文。

注意

赋值给事件驱动通讯块的背景 DB 的报文文本只能在 CFC 中编辑。

14.3 设定和编辑与符号相关的信息

14.3.1 设定和编辑与符号相关的信息

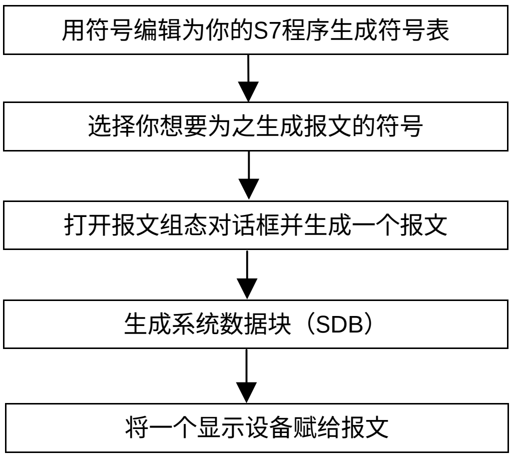
在符号表中符号相关的报文（SCAN）被直接赋值给信号。所允许的信号都是布尔地址：输入（I）、输出（Q）及位存储（M）。你可以用报文组态功能给这些信号赋予不同的属性、报文文本及多达 10 个的相关值。你可以通过设置筛选在符号表中轻松地选择信号。

用符号相关报文，你可以以一个预先设定的时间间隔来扫描信号，从而决定是否有信号变化。

注意

时间间隔依据所用的 CPU 而定：

基本程序



在程序处理过程中，与程序同步对已组态报文的信号进行检查。这种检查以组态的时间间隔进行。报文显示在指定的显示设备上。

14.4 生成并编辑用户定义的诊断报文

14.4.1 生成并编辑用户定义的诊断报文

使用这一功能，你可以在诊断缓冲区中写入用户条款，并发送在报文组态应用程序中生成的相应的报文。用户定义的诊断报文可通过用作报文块的系统功能 SFC52（WR_USMSG）生成。你必须在用户程序中插入对 SFC52 的调用并给它分配事件 ID。

与块相关及符号相关报文相比，用户定义的诊断报文只能显示在编程设备上。因此，你不能在报文组态应用程序中将显示设备赋给这些报文。

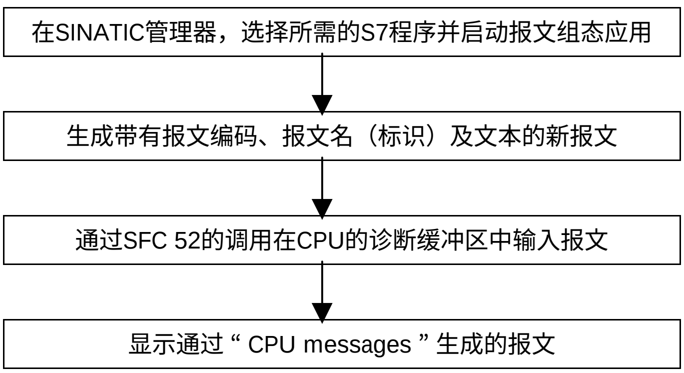
要求

在生成用户定义的诊断报文之前，必完成以下事项：

- 在SIMATIC管理器中生成项目
- 在项目中生成S7/M7程序，报文将被分配给该程序

基本程序

成并显示用户定义的诊断报文，可按以下步骤进行：



14.5 翻译并编辑用户文本

14.5.1 翻译并编辑操作文本

在过程编辑当中输出到显示设备上的文本通常用与自动化任务编程相同的语言输入。

也许会有这样的情况，对显示设备上的报文进行反应的操作员却不讲这种语言。这样的用户需要的是他自己语言的文本，以确保顺利、无障碍地处理并对系统输出的报文快速反应。

STEP 7 允许你将任意的、所有操作相关的文本翻译成所需的语言。唯一的前提条件是你已在项目中安装了该语言（SIMATIC 管理器中的菜单命令 Options> Language for Display Devices）。可用的语言的数量由安装 Windows 时决定（系统特性）。

用这种方法你可以确信，以后任何用户面对这条报文都可以让它以适当的语言显示。这一系统特征明显地增加了过程的安全性和准确性。

操作者相关文本为用户文本和文本库。

14.5.2 翻译并编辑用户文本

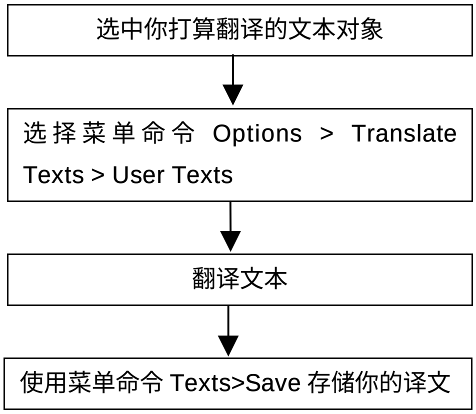
对整个项目、S7 程序、块文件夹、单个块或符号表来说，如果在这些对象中有组态的报文，则可生成用户文本列表。它们包含有比如所有文本和报文，可以在设备上显示。对于一个项目，可以有一些操作相关文本列表，你可以将它们翻译成所需的语言。

你可以在一个项目中选择已有的语言（菜单命令：Options > Language for Display Devices…）。还可以在晚些时候增加或删除语言。

当你打开 STEP 7 对象的用户文本（菜单命令 Options > Translate Texts > User Texts）时，在显示屏上将显示一个表。其中每种语言一栏。第一栏中列表中总是显示缺省设置的语言。

基本程序

在 SIMATIC 管理器中，用菜单命令 Options>Language for Display Devices...，设置好了你的用户文本想要转换的语言。



导出和导入操作相关的文本

你可以翻译或编辑相关的操作文本，无论它是在 STEP 7 内生成的还是在 STEP 7 之外生成的。要实现这一点，将所显示的相关的操作文本列表导出到一个 CSV 文本文件中，这个文本文件可以在 ASCII 编辑器或象 Microsoft Excel 这样的表格编辑器中编辑。然后你可将文本导回 STEP 7。

相关的操作文本从项目哪个部分导出的，只能从这个部分导入。

14.5.3 翻译文本库

文本库提供有可以集成在报文中的文本列表，并可在运行时间动态刷新，显示在编程器或其它显示装置上。并将赋值给 CPU。系统文本库中的文本可以由 STEP 7 或 STEP 7 任选软件包提供。可以有几个文本库赋值给一个 CPU。你可将这些文本翻译成所需语言。

在 SIMATIC Manager 中，你可以选择适用于项目的语言（菜单命令 Options > Language for Display Devices...）。以后，你也可以添加或删除语言。

当你打开一个文本库时（菜单命令 Options > Translate Texts > System Text Library），在显示屏上将显示一个表。表的每一栏都表示一种语言。第一栏总是显示你可引用文本的索引。

举例

索引	德语	英语
1732	ausgefallen	failed
1733	gestört	faulty
1734	Parametrierfehler	parameter assignment error

基本程序

在 SIMATIC Manager（SIMATIC 管理器）中，用菜单命令 Options > Language for Display Devices...，设置好了你的用户文本想要转换的语言。

选中你打算翻译文本对象的 CPU 或 CPU 下的一个对象

↓

选择菜单命令 Options > Translate Texts > System Text Library，并选择你要编辑的文本

↓

翻译文本

↓

使用菜单命令 Texts > Save 存储你的译文

14.6 传送报文组态数据到可编程控制器

14.6.1 传送报文组态数据到可编程控制器

概述

使用传送程序 PLC-OS Engineering，你可将生成的报文组态数据送至 WinCC 数据库。

你可以选择许多不同的传送选项。比如，可以选择一个地址和文本比较以确保当前数据已被传送。

要求

在开始传送之前，下列要求必须满足：

- 已安装了PLC-OS连接组态的Setup程序。
- 已为建立报文生成了组态数据

基本程序

在 SIMATIC 管理器中，生成你想要显示报文的 OS 对象

↓

使用菜单命令 Options > PLC-OS Connection Data > Transfer，启动组态数据的传送

14.7 显示 CPU 报文和用户定义的诊断报文

14.7.1 显示 CPU 报文和用户定义的诊断报文

用“ CPU Message（CPU 报文）”功能，可以显示诊断事件的异步报文和用户定义的诊断报文。

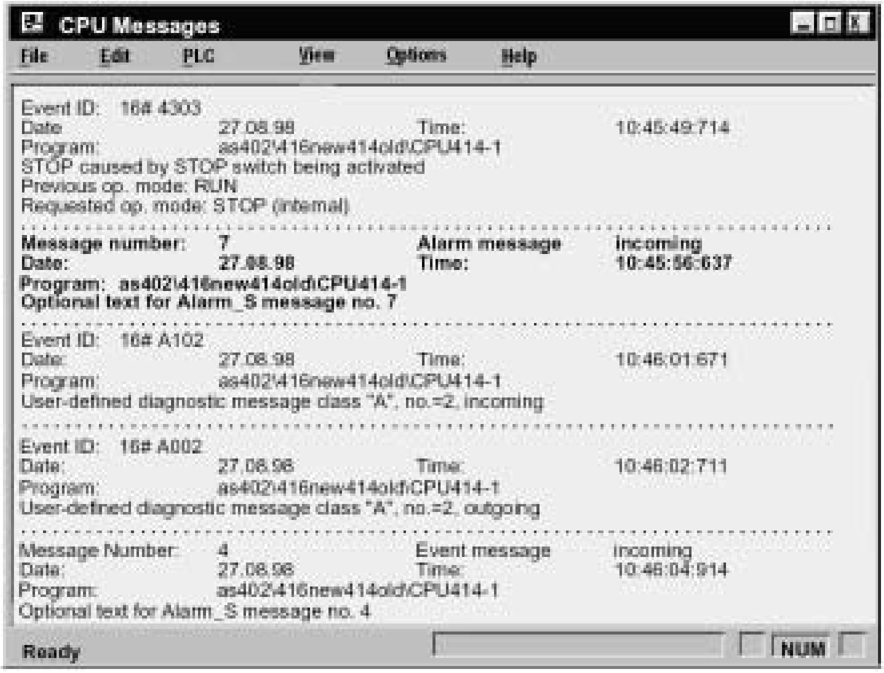
你还可以通过使用菜单命令 Options>Configure Messages 从 CPU Messages 应用程序中启动报文组态应用程序，并且生成用户定义的诊断报文。这就要求你通过一个在线项目启动 CPU Messages 应用程序。

显示选项

用“ CPU Messages ”功能，可以决定所选 CPU 的在线报文是否显示以及如何显示：

- “ Bring to the Foreground（至前台）”：包含CPU报文的窗口显示在前台。每当收到新报文，该窗口都显示于屏幕的前台。
- “ Leave in the Background（至后台）”：CPU报文被接收到后台中。当收到新报文时该窗口仍处于后台中，如果需要可以被提到前台。
- “ Ignore Message ”：CPU报文不显示，并且与前两种模式相比，不存档。

在“ CPU Messages ”窗口，你可以在存档中浏览报文。下图为若干示例：



可应答报文（ALARM_SQ），显示为黑体字并且可以使用菜单命令 Edit > Acknowledge CPU Message 进行应答。

存档功能

有一个用以备份报文的档案（文件），其中可存储 40 至 2000 条 CPU 报文。如果超过了设置的存档容量，档案中最旧的报文被删除以便留出空间接收新报文。

刷新ALARM_S/ALARM_SQ报文

当 ALARM_S/SQ 报文被刷新时，所有未发出及/或未应答报文再次被送入档案。报文被刷新：

- 如果与报文相关的模板执行了再起动（不是冷起动）。
- 如果你组态CPU报文时在 “ Customize ” 对话框中为ALARM_S/SQ选中选项 “ A ”。

基本程序

为所选的模板组态 CPU 报文。

在 SIMATIC 管理器中选择 S7 Program 并选择菜单命令 PLC > CPU Messages。

↓

确定要接收的报文，并确定如何显示它们。

14.7.2 组态 CPU 报文

要为所选择的模板组态 CPU 报文，可如下进行：

1. 在 SIMATIC 管理器中，通过一个在线项目起动 CPU 报文应用程序。要实现这一步，选择一个在线在 S7 程序并且使用菜单命令 PLC>CPU Messages 为所选的 CPU 调用报文应用程序。

   结果：出现“ CPU Messages ”应用窗口，在此列出了所记录的 CPU。

2. 你可以通过重复步骤 1.为其它程序或接口扩展注册 CPU 列表。

3. 在列表选项前点击复选框，并指定应为该模板接收哪些报文：

   A：激活（ALARM_SQ （SFC 17）和 ALARM_S （SFC 18），例如，报告 S7 PDIAG、 S7-GRAPH 过程诊断信息或系统错误。

   W：激活诊断事件。

4. 设置档案的容量。

   结果：只要上述报文一出现，它们就被写入报文档案并且以你所选择的方式显示。

注意

你在 SIMATIC 管理器中使用菜单命令 PLC>CPU Messages，调用的 CPU 在“ Customize ”应用窗口中已被存入注册模板列表。列表中的各项除非在“ CPU Messages ”应用窗口中删除，否则会一直保留下来。

14.7.3 显示存储的 CPU 报文

CPU 报文总会被记录在档案中，除非你选择了菜单命令“ View > Ignore Message ”。所有的存档报文总是会被显示。

14.8 组态“ 系统错误报告 ”

14.8.1 组态“ 系统错误报告 ”

简介

当系统错误时，S7 组件和 DP 标准从站可以激活组织块调用。

例如：如果有断线，具有诊断功能的模块就可以触发诊断中断（OB82）。

S7 组件可以提供系统错误信息。开始事件信息，即赋值 OB 的局域数据（包括数据记录 0）提供有关错误的位置（例如模块的逻辑地址）和类型（例如通道错误或备份错误）的一般信息。

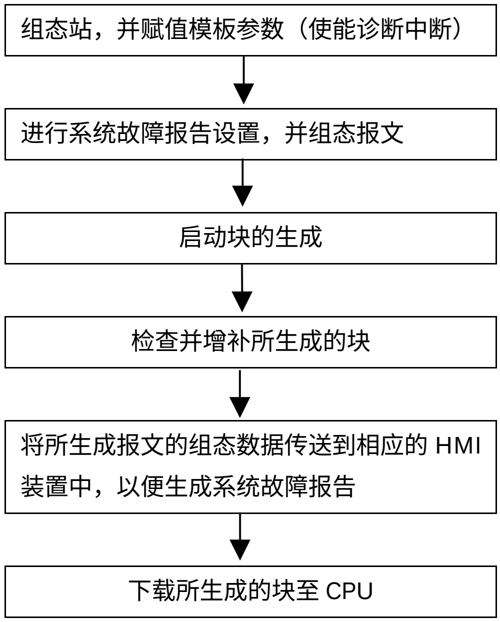
另外，错误还可以详述其它诊断信息（使用 SFC51 读取数据记录 1 或使用 SFC13 读取 DP 标准从站的诊断报文）。例如通道 0 或通道 1 以及断线或测量范围超限错误。

使用“ Report System Error（报告系统错误） ”功能，STEP 7 提供了一种极其方便的由报文形式的组件提供的诊断信息显示方法。

STEP7 可以自动生成所需块和报文文本。所有用户所需要做的就是将所生成的块装入 CPU 中，并将文本传送至所连接的 HMI 设备。

你可查找“ Supported Components（支持组件） ”和“ Functional Scope（功能范围） ”部分中各种从站的诊断信息总览。

基本程序



通过标准报文路径 ALARM_S/SQ，将报文传送至编程器上的 CPU Messages 中或所连接的 HMI 装置。

14.8.2 支持组件和功能范围

只要 S7 300 站、S7 400 站和 DP 从站支持诊断、中断、插入/移开模板中断以及通道诊断等功能，“ Report System Error（报告系统错误）”就支持它们。

下述组态不被“ Report System Error（报告系统错误）”所支持：

- S7-300站中的DP主站接口模板（CP 342-5 DP）上的M7、C7和PROFIBUS-DP组态。
- H系统。
- SIMATIC PC站（软件PLC、插槽式PLC）。

在重新启动时，你必须注意可能会出现遗失的中断报文。这是由于 CPU 的报文响应存储器在重新启动时，没有删除，但 Report System Error 重新设置的内部数据。

在下面两个表中，你可找到 Report System Error 所支持的各种从站的所有诊断块。

诊断块	ID(故障槽)	通道名称(通道错误)[1)]	模板状态（模板错误、不正确/没有模板）	制造商名称	设备名(只适用于 ET 200 B)
Header ID 2）	0x01	0x10	0x00 Type 0x82	0x00 Type > 0x9F	0x00 + 1 Byte Diagnostic Info
ET 200S	报文：“诊断可用”	纯文本报文	纯文本报文	.GSE 文件纯文本报文 3）	- 4）
ET 200M 作为 S7 从站	不评估	不评估	不评估	-	-
ET 200M 作为标准从站	报文：“诊断可用”	纯文本报文	纯文本报文	-	-
ET 200X	报文：“诊断可用”	-	-	-	-
ET 200X DESINA	报文：“诊断可用”	纯文本报文	纯文本报文	-	-
ET 200L	报文：“诊断可用”	-		-	-
数字 ET 200B	报文：“诊断可用”	-	-	-	报文：“模板故障”
模拟 ET 200B	报文：“诊断可用”	-	-	-	

1） DPV0 后的 GSE 文本标准报文。
2） 标题标识符：诊断报文中可以标识不同诊断部分的标识符。
3） GSE 文件：DPV0 或 DPV1 Norm 从站文件形式描述
4） “ - ”意思是组件不提供信息。

诊断块	DS0/DS1 1	H 状态 (H 系统目前还有支持)	其它实例
Header ID 2	0x00 Type 0x01	0x00 Type = 0x9E 或 Type = 0x9F	0x00 其它类型
ET 200S	纯文本报文	-...3)	不评估
ET 200M 作为 S7 从站	纯文本报文	不评估	不评估
ET 200M 作为标准从站	纯文本报文	不评估	不评估
ET 200X	报文："模板故障"	-	不评估
ET 200X DESINA	纯文本报文	-	不评估
ET 200L	报文："模板故障"	-	-
数字 ET 200B	-	-	-
模拟 ET 200B	纯文本报文	-	不评估

1）DS0：标准诊断，例如模板错误、外部辅助电压或正面连接器遗失、范围 4 字节、OB82 局域数据中。

DS1：通道错误，每种通道类型都不同，可通过 SFC 51 在用户程序中读取。文本来自 S7 HW 诊断。

2）标题标识符：诊断报文中可以标识不同诊断部分的标识符。

3）"-"意思是组件不提供信息。

诊断报文（又成为 Norm 从站报文）有上述诊断块组成，并可通过 SFC 13 使用用户程序读取。

在 STEP 7 中，在"Hex display"下，通过在"DP Slave Diagnostics"选项卡中的在线窗口"HWConfig"中调用模块状态，可以显示诊断报文。

14.8.3 报告系统出错设置

有几种方法可调用设置对话框：

- 在HW Config中，选择你想组态系统错误报告的CPU。选择菜单命令Options > Report System Error。
- 如果已经生成报告系统错误块，双击所生成的块（FB、DB），可以调用对话框。
- 在站的"Properties（属性）"对话框中，在Save（保存）和Compile（编译）组态时，选择自动调用选项。

在 Save（保存）和 Compile（编译）时，你可以如下进入自动调用选项：

1. 在 SIMATIC 管理器中，选择相应的站。
2. 选择菜单命令 Edit＞Object Properties。
3. 选择“ Settings ”选项卡。

在对话框中，输入以下内容：

- 生成FB以及赋值背景DB。
- CPU出错特性：你可以设置在系统出错报告后，是否将CPU切换为“ STOP ”。
- 生成错误OB：在S7程序中是否应生成仍不能使用的错误OB。
- 是否应响应报文。
- 是否应生成参考数据。
- 是否在生成Report System Error（报告系统错误）时，总是显示警告。
- 是否在保存和编译组态后（见上述设置）自动调用Report System Error时，显示对话框。
- 报文外部特征（文本部分的结构和顺序）

在调用对话框上的帮助中，你会得到更详细的信息。

14.8.4 生成报告系统出错块

为了生成所需的块（FB 和 DB，并根据设置，OB 仍不可用），如下进行：

- 在“ Report System Errors ”对话框中，点击“ Generate（生成） ”按钮。

可以生成以下块：

- 错误OB（如果“ Generate error OBs（生成错误OB） ”复选框激活）。
- 诊断FB（缺省：FB49）。
- 诊断FB背景DB（缺省：DB49）。
- 由诊断FB调用的任选用户FB。

14.8.5 生成错误 OB

如果在“ Report System Error ”对话框的“ General（一般） ”任选卡中，激活复选框“ Generate error OBs（生成错误 OB） ”：

- OB81（电源故障），调用所生成的诊断FB。
- OB82（诊断中断OB；仅适用于组态模板DP从站），调用所生成的诊断FB。
- OB83（插入/清除中断），调用所生成的诊断FB。
- OB84（CPU硬件故障）该OB生成时没有内容，因此，当通讯发生故障时，CPU 不会切换到 STOP 模式（例如当插入或移开 MPI 电缆时，MPI 终端电阻故障）。对错误不评估；不生成报文。
- OB85（程序执行错误） 如果在刷新过程映象时（例如移开模板时）出现错误，只能防止CPU 切换到STOP 模式。由此，可以处理OB83中的诊断FB。任何一个Report System Error 报文后的CPU STOP设置，都会影响OB83。对于所有其它OB85错误事件，CPU都将进入STOP模式。
- OB86（扩展机架、DP主站系统或分布式I/O设备故障），可以调用生成的诊断FB。 该OB只有在组态上述组件之一时，才生成。

如果故障OB已存在...

现有故障 OB 不会被覆盖。你必须亲自在相应 OB 中插入调用所生成的 FB。

如果组态包括分布I/O设备...

对于分布 I/O 中的评估错误，所生成的 FB 会自动调用 SFC13（读取 DP 从站的诊断数据）。为了保证该功能，所生成的 FB 必须在 OB1 或短时循环中断 OB 以及启动 OB 中调用。

注意

请注意以下事项：

- 当由于“ Error While Updating Process Image（刷新过程映象时出现的错误）”错误事件， Report System Error生成OB85时，CPU不能进入STOP模式。
- 当出现以下错误时，CPU还可调用OB85：
  - OB“ That Is Not Loaded Error Event（没有装入 OB 错误事件） ”
  - “ Error When Calling or Accessing an OB That Is Not Loaded（调用或访问没有装入的 OB 错误） ”

当出现这些错误时，如果 Report System Error 在使用之前生成 OB85，CPU 仍可进入 STOP 模式。

- 由于在这些OB中不会调用Report System Error的FB，在执行诊断FB后，CPU进入STOP模式不会影响OB84和OB85。对于OB85，该设置可以通过在OB83中调用FB直接说明。

14.8.6 所生成的 FB、DB

所生成的 FB 可以评估故障 OB 的局域数据，读取触发故障的 S7 组件的其它诊断信息，并自动生成相应的报文。

FB 具有以下特性：

- RSE（Report System Error，报告系统错误）生成语言（也适用于所生成的背景DB）。
- Know-how保护（也适用于所生成的背景DB）。
- 运行期中断延时。
- 双击调用 “ Report System Error ” 功能设置对话框（也适用于所生成的背景DB）。

用户块

由于诊断 FB 为 know-how 保护，因此你不能对它进行编辑。但是，FB 提供有用户程序接口，因此你可以存取错误状态或报文编号等。

使用以下参数可以在所生成的 FB 中调用用户程序中的评估块（可以在对话框中的 “ User Block ” 选项卡中设置）：

```
EV_C：BOOL；                // 来报（TRUE）或外发报文（FALSE）
EV_ID：DWORD ；             // 生成报文编号
IO_Flag：BYTE ；            // 输入模板：B#16#54  输出模板：B#16#55
logAdr：WORD；              // 逻辑地址
TextListID：WORD；          // 文本列表 ID（缺省列表= 1）
ErrorNo：WORD ；            // 生成故障编号
Channel_Error：BOOL ；      // 通道错误（TRUE）
ChannelNo：WORD；           // 通道编号
```

如果没有用户 FB，可以使用上述参数由 SFM 生成。

15 控制和监视变量

15.1 组态操作员控制和监视变量

概述

当你使用 WinCC 对过程或可编程控制器进行监控时，STEP 7 可提供用户友好方式对变量进行控制和监视。

与前面的方法相比，这一方法的优势在于你不需要再去给每一个操作站（OS）分别作组态数据，你只需要使用 STEP 7 作一次组态即可。你可以使用传送程序 PLC-OS Engineering（“ Process Control Syetem PCS 7 ” 软件包的一部分）将 STEP 7 中组态生成的数据传送给 WinCC，在传送过程中要检查数据的一致性及它们与显示系统的可兼容性。WinCC 使用变量块及图象对象中的数据。

使 STEP 7，你可以为下列变量组态或修改操作员控制和监视属性：

- 功能块中的输入、输出及入/出参数
- 存储位及I/O信号
- CFC图表中的CFC块的参数

基本程序

组态操作员控制及监视变量的基本程序要依据所选的编程/组态语言以及想控制和监视的变量而定。但基本程序总会包括以下各步：

1. 为操作员控制及监视将系统属性赋给功能块的参数或赋给符号赋值表中的符号。
   在 CFC 中无需此步骤，因为你可以使用库中已准备好的块。
2. 给你想控制和监视的变量赋予所需的属性在一个对话框中的（S7_m_c）登录特性。在“ Operator Interface（操作者接口）”对话框中（菜单命令Edit > Special Object Properties > Operator Interface），你可以改变WinCC属性，例如极限值、替代值以及协议特性等。
3. 使用工具PLC-OS Engineering将STEP 7中生成的组态数据传送到你的显示系统（WinCC）。

命名规则

为存储和传送而给 WinCC 作的组态数据要存储在一个由 STEP 7 自动赋赋给的一个唯一的名字下面。用于操作员控制及监视、CFC 图表以及 S7 程序的变量名是组成该名字的一部分，因此要符合以下规律：

- 在S7项目下的S7程序的名字要唯一（不同的站不能包含同名的S7程序）。
- 变量、S7程序和CFC图表的名字不能包含下划线、空格或下列特殊字符： [‘] [.] [%] [-] [/] [*] [+]。

15.2 用语句表、梯形图及功能块图组态操作员控制及监视的属性

15.2.1 用语句表、梯形图及功能块图组态操作员控制及监视的属性

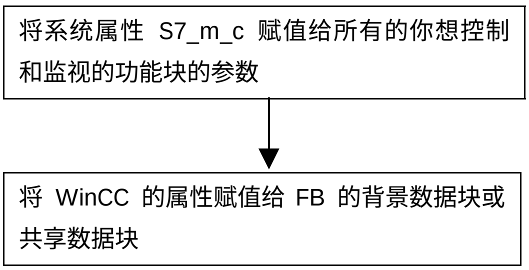
概述

使用下面描述的步骤，你可以设置功能块的参数以适合于操作员控制和监视，并且可以将所需的 O，C，M 属性赋值给你的用户程序中相关的背景数据块或其享数据块。

要求

你必须有一个 STEP 7 项目，一个 S7 程序以及一个功能块。

基本程序



15.3 用符号表组成控制与监测

15.3.1 用符号表组成控制与监测

概述

不论使用什么编程语言，你都可以按下面所描述的步骤态下列变量：

- 位存储
- I/O信号

要求

在你开始之前，必须满足下列要求：

- 在SIMATIC管理器中已生成一个项目。
- 在该项目下有一个带有符号表的S7程序。
- 该符号表必须是打开的。

基本程序

将特殊对象特性“ Operator Control and Monitoring（操作员控制和监视）”赋给符号

↓

将 WinCC 属性赋给符号

15.4 用 CFC 改变操作员控制和监视属性

15.4.1 用 CFC 改变操作员控制和监视属性

概述

在 CFC 中，你可以从库中选择已有操作员控制和监视功能的块来生成你的用户程序，并且将这些块在图表中相应的位置上相连接。

要求

在 STEP 7 项目中已插入一个 S7 程序，生成了 CFC 图表并放置块在里面。

基本程序

编辑块的对象特性

注意

如果你使用自己生成的块，并且已给它们赋予了系统属性 S7_m_c，你可以通过激活“Operator Control and Monitoring”对话框中的“Operator Control and Monitoring”复选窗口给这些块赋予操作员控制和监视能力（菜单命令 Edit > Special Object Properties > Operator Control and Monitoring）。

15.5 传送组态数据到操作员接口可编程控制器

15.5.1 传送组态数据到操作员接口可编程控制器

概述

使用传送程序 PLC-OS Engineering，你可以将为操作员控制和监视而生成的组态数据送到 WinCC 的数据库中。

你可以选择许多不同的传送选项。比如，可以选择一个地址和文本比较以确保当前 WinCC 属性已被传送。

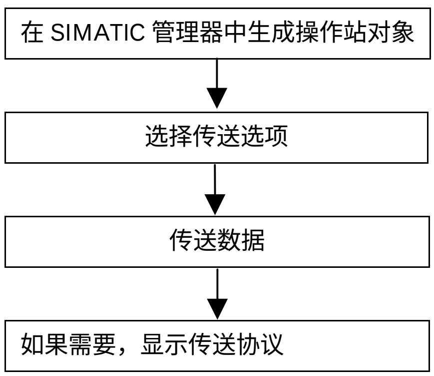
要求

在开始传送之前，下列要求必须满足：

- 已经安装了PLC-OS连接组态的Setup（建立）程序（Engineering）。
- 已为操作员控制和监视生成了组态数据。

基本程序

将操作员控制和监视的组态数据传送到 WinCC 数据库，其过程如下：



16 建立在线连接进行 CPU 设置

16.1 建立在线连接

16.1.1 建立在线连接

要下载 S7 用户程序/块、从 S7 可编程控制器上载程序到编程器和其它操作，需在编程设备和可编程控制器之间建立在线连接：

- 测试用户程度
- 显示和改变CPU操作模式
- 为CPU设置时间和日期
- 显示模板信息
- 比较在线和离线的块
- 诊断硬件

为建立在线连接，编程设备和可编程控制器必须通过合适的接口相连接（例如，多点接口（MPI））。然后，你可以通过一个在线的项目窗口或“ Accessible Nodes（可访问站） ” 窗口访问可编程控制器。

16.1.2 通过 “ Accessible Nodes ” 窗口建立在线连接

这种访问方式可令你快速访问可编程控制器，可用于测试的目的。你可以在网络中访问所有的可访问的可编程模板。如果你的编程设备中没有关于可编程控制器的项目数据则可选择该种方式。

使用菜单命令 PLC>Display Accessible Nodes，打开 “ Accessible Nodes ” 窗口。在 “ Accessible Nodes ” 对象中显示网络中所有可访问的可编程模板及其地址。

那些不能用 STEP 7 编程的站（如编程设备或操作面板）也能显示出来。

16.1.3 通过在线的项目窗口建立在线连接

如果在编程设备/PC 的项目中有组态的可编程控制器，你可以选择该种方法。你可以在 SIMATIC 管理器中使用菜单命令 View>Online 打开一个在线窗口。该窗口显示可编程控制器中的项目数据（离线窗口与之相反，显示编程设备/PC 中的项目数据）。可编程控制器中的 S7 程序和 M7 程序的数据都可以在在线窗口显示。

你可以将项目的这种显示用于与访问可编程控制器相关的功能。SIMATIC 管理器中，菜单“ PLC ”中的某些功能只能在在线窗口中激活，不能在离线窗口中使用。

有以下两种访问类型：

- 用组态的硬件访问

  这意味着你只能访问离线组态的模板。哪个在线模板可以访问由组态可编程模板时设置的 MPI 地址决定。

- 无需组态硬件的访问

  这种方法要求有一个与硬件无关的 S7 程序或 M7 程序存在（即该程序直接位于项目之下）。由 S7/M7 程序对象属性中定义的相应的 MPI 地址决定哪个在线模板可以访问。

通过在线窗口访问，显示的数据是可编程控制器与编程设备中的数据的组合。比如，如果你在一个在线项目下打开一个 S7 块，则显示组合如下：

- 来自于可编程控制器CPU中的块的指令代码部分，及
- 来自编程设备数据库中的注释和符号（如果离线程序中有这些注释和符号）。如果你不通过项目结构，而是直接打开所连的CPU，则显示的程序就是这些程序在CPU中的原样，即没有符号和注释。

16.1.4 设定访问可编程控制器的口令

使用口令保护，你可以：

- 保持CPU中的用户程序及其数据未经授权不能改变（写保护）
- 保护你的用户程序中的编程专利（读保护）
- 保护在线功能以免可能对过程造成干扰

如果模板支持口令保护的功能，你用这种方法就可以保护该模板。

如果你想用一个口令来保护一个模板，则必须在对模板进行参数赋值的过程中定义保护级别并设置口令，然后将修改的参数送至模板。

如果需要对执行在线功能输入口令，“ Enter Password ” 对话框会显示出来。若输入正确的口令，你就可以得到模板的访问权，该模板在参数赋值过程中已被设置了特定的保护级别。此时你就可以与被保护的模板建立在线连接并执行属于那个保护级别的在线功能。

使用菜单命令 PLC>Access Rights> Setup，你可以直接调用“ Enter Password ” 对话框。由此，例如，在对话开始时，你可以一次输入口令，以后在线访问时，将不再询问。输入的口令将有效至 SIMATIC Manager 关闭或使用菜单命令 PLC > Access Rights > Cancel 取消口令。

CPU 参数	要点
测试操作/过程操作（无 S7-400 或 CPU 318-2）	可在 “ Protection ” 选项卡中设置 在过程操作中，为保证设置允许的循环扫描时间的增加不超限，象程序测试或监视／修改变量这样的测试功能是受制约的。 这意味着，比如，在程序测试中不允许调用条件，并且编程循环的状态显示在返回点被中断。 使用断点的测试及程序的单步执行在过程操作中不能使用。 在测试操作中，通过编程设备/PC 的所有测试功能都可以不受限制地使用，即使这些功能可能会使循环扫描时间的增加相当可观。
保护级别	可在“ Protection ”选项卡中设置如果知道正确的口令，你就可以对 CPU 进行写读/写访问。口令在此选项卡中设定。

16.1.5 刷新窗口内容

你需要注意以下事项：

- 由于用户操作（如下载或删除块）而在一个在线的项目窗口造成的修改不会在任何已打开的“ Accessible Nodes（可访问站）”窗口自动刷新。
- 任何在“ Accessible Nodes ”窗口中所作的修改，不会自动刷新任何已打开的在线项目窗口。

要刷新一个并行打开的窗口，你必须直接在该窗口刷新（使用菜单命令或功能键 F5）。

16.2 显示和改变操作模式

16.2.1 显示和改变操作模式

例如用这个功能，你能够在纠正错误之后将 CPU 切换到 “ RUN ” 。

显示操作模式

1. 打开你的项目并选择 S7/M7 程序，或用菜单命令 PLC>Display Accessible Nodes 打开 “ Accessible Nodes ” 窗口并选择一个站（ “ MPI=… ” ） 。
2. 选择菜单命令 PLC>Operating Mode。

该对话框显示当前和最近一次的操作模式以及在模板上当前模式选择开关的设置。对于那些无法显示其当前开关设置的模板，显示文本 “ Undefined ” 。

改变操作模式

你可以使用按钮改变 CPU 的模式。只有当这些按钮是激活的，才能在当前的操作模式下被选择。

16.3 显示并设置时间和日期

16.3.1 显示并设置时间和日期

按如下进行：

1. 打开你的项目并选择S7/M7程序，用菜单命令PLC>Display Accessible Nodes打开“Accessible Nodes”窗口并选择一个站（“MPI=…”）。
2. 选择菜单命令PLC>Set Date and Time。

   只有当所选的 S7/M7 程序是在一个项目窗口（在线显示）或在“Accessible Nodes”窗口选中了一个站（“MPI=...”），该命令才能被选中。
3. 在显示的对话框中你可以读到所选模板当前的时间和日期。
4. 如果需要，可以在“Date”（日期）和“Time”（时间）栏中输入新的值，或者用缺省选项接受你的编程设备/PC上的时间和日期。

注意

如果模板没有实时时钟，对话框中时间显示为“00：00：00”，日期显示为“00.00.00”。

---

17 下载

17.1 从 PG/PC 中下载到可编程序控制器

17.1.1 下载条件

下载到可编程序控制器的条件

- 你的编程设备和可编程序控制器里的CPU之间必须有一个联接（例如，多点接口）
- 必须可以访问可编程序控制器。
- 你下载的程序已无误地编译。
- CPU必须在允许下载的工作模式下（STOP或RUN-P）。

  RUN-P 模式表示，这个程序将一次下载一个块。如果你这样重写一个旧的 CPU 程序，可能出现冲突，例如，如果块参数已改变了。当处理循环时，CPU 就会进入 STOP 模式。我们因此建议你在下载前将 CPU 切换到 STOP 模式。
- 如果你要离线打开一个块并要下载它，CPU必须与SIMATIC管理器中的一个在线用户程序相联接。
- 在下载你的用户程序之前，你必须使CPU复位，保证没有旧块在CPU上。

STOP方式

在你做下列事情前，把工作方式从“ RUN ”模式置到“ STOP ”模式：

- 下载完整的或部分用户程序到CPU
- 在CPU执行存储器全清
- 压缩用户存储器

热启动（转换到“ RUN ”方式）

如果你在“ STOP ”模式时执行一个热启动，程序重新启动并在“ STARTUP ”模式下首先运行启动程序（在块 OB100 里）。如果启动成功，CPU 转到“ RUN ”方式。做下列事后需要热启动：

- CPU复位
- 在STOP模式下载用户程序

17.1.2 保存和下载块的区别

你一定要区分保存块和下载块。

	保　存	下　载
菜单命令	File > Save File > Save As	PLC > Download
功能	编辑器中块的当前状态被保存在编程设备的硬盘上	编辑器里块的当前状态仅下载到 CPU 上
语法检查	运行语法检查。任何错误都将在对话框里被报告。错误的原因和位置也将显示出来。在你下载或保存块前必须纠正这些错误。如果在语法上没有发现错误，块将被编译成机器码或者保存下载。	运行语法检查。任何错误都将在对话框里被报告。错误的原因和位置也将显示出来。在你下载或保存块前必须纠正这些错误。如果在语法上没有发现错误，块将被编译成机器码或者保存下载。

这个表的应用与你是在线打开块还是离线打开块无关。

块修改的技巧——先保存后下载

要存入新创建的块或在逻辑块的程序区中进行修改，在声明表中修改或在数据块中输入新的或更改的数据值，你必须对各个块进行保存。用命令菜单 PLC>Download 可将在编辑器里进行的任何修改传送到 CPU，例如，检查小的修改—，在你退出编辑程序前，任何情况下都必须（把变更）保存在编程设备的硬盘上。否则，在 CPU 里和编程设备上，你将得到不同版本的用户程序。一般建议你先保存所有的更改，然后再下载它们。

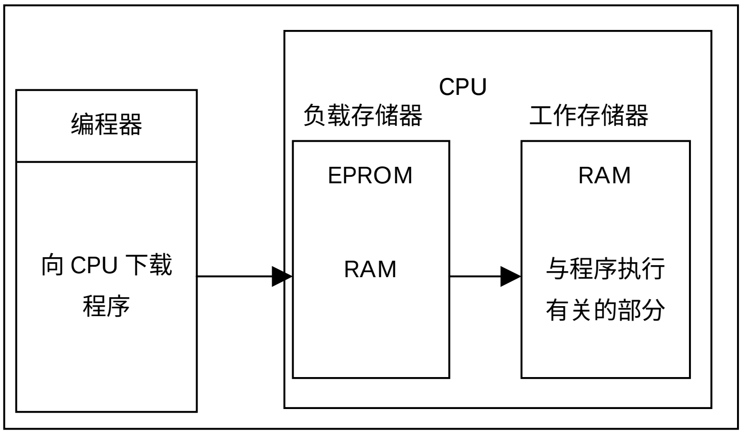
17.1.3 CPU 里的装载存储器和工作存储器

在完成组态，参数赋值，程序创建和建立在线连接后，你可以下载整个用户程序或个别块到一个可编程序控制器。要测试单个块，你必须至少下载一个组织块（OB），在 OB 里被调用的功能块（FB）和功能（FC）以及使用的数据块（DB）。要下载硬件组态，网络组态和创建连接表生成的系统数据到可编程控制器，你可下载对象 “ System Data ” 。

在程序测试的末段或运行已完成了用户程序期间，你可以用 SIMATIC 管理器下载用户程序到一个可编程序控制器。

关系—装载存储器和工作存储器

完整的用户程序下载到装载存储器；与程序执行有关的部分也已经装入到工作存储器中。



CPU 装载存储器

- 装载存储器用来存储没有符号表和注释的用户程序（这些符号和注释保留在编程设备存储器中）
- 没有被标记为运行需要的块，将仅存储在装载存储器里
- 装载存储器可以是RAM，ROM或EPROM存储器，随可编程序控制器而定
- 装载存储器也可以有一个集成的EEPROM部分以及一个集成的RAM部分（如，CPU 312IFM和CPU 314IFM）
- 在S7-400中，用一个存储卡（RAM或EEPROM）来扩展装载存储器是很必要的

CPU工作存储器

工作存储器（集成 RAM）用来存储程序处理所需要的那一部分用户程序。

可能的下载/上载步骤

- 你用下载功能下载用户程序或可装载的对象（如，块）到可编程序控制器。如果这个块已存在于CPU的RAM中，你将被注意，要确定这个块是否要重写。
- 你可以在项目窗口中选择可装载的对象，并从SIMATIC管理器中下载它们（菜单命令：PLC > Download）。
- 当编程块，组态硬件和网络时，可直接下载这个当前正在编辑的对象，你可以用你工作所需的应用程序的主窗口里的菜单来进行。（菜单命令：PLC > Download）。
- 另一个可能是，打开一个在线窗口查看可编程控制器（例如，用View>Online或PLC>Display Accessible Nodes），并且复制你想要下载到这个在线窗口的对象。

另外可以通过装载功能从 CPU 的装载存储器 RAM 把块的当前内容上载到你的可编程序控制器里。

17.1.4 下载方式随装载存储器而定

CPU 里装载存储器中 RAM 区和 EEPROM 区的分配决定了在你的用户程序中下载用户程序或块时可用的方式。以下是下载数据到 CPU 中的可能方式：

装载存储器	装载方式	PG 和 PLC 之间的通信类型
RAM	下载和删除各个块	PG-PLC 在线连接
	下载和删除整个用户程序	PG-PLC 在线连接
	重新装入各个块	PG-PLC 在线连接
集成（仅 S7-300）或外插 EPROM	下载整个用户程序	PG-PLC 在线连接
外插 EPROM	下载整个用户程序	外部加载 EPROM 并插入存储卡或通过插入 EPROM 的 PLC 上的在线连接。

通过在线连接下载到RAM

在可编程序控制器中，如果掉电且 RAM 没有被备份则数据丢失。这种情况下，RAM 中的数据也将随着丢失。

保存到EPROM存储卡中

块或用户程序被保存在一个 EPROM 存储器上，然后插在 CPU 的一个插槽中。

存储卡是便携式数据记录媒体。它们通过编程设备来写入，然后插在 CPU 上恰当的插槽里。

当电源关断后和 CPU 复位时，保存在它们上面的数据将被保留。在 CPU 存储器复位且电源掉电之后，电源重新恢复时如果 RAM 中的内容没有备份，EPROM 中的内容被重新复制到 CPU 存储器的 RAM 区。

保存在集成EPROM里

对于 CPU312，你也可以保存 RAM 的内容到集成 EPROM 中。在电源断开时，集成的 EPROM 中的数据将被保留。在电源断开且 CPU 存储器复位后再重新恢复时，如果 RAM 没有被备份，集成的 EPROM 中的内容将被重新复制到 CPU 存储器的 RAM 区。

17.1.5 在可编程控制器中重新下载块

对 S7 可编程控制器，CPU 装载存储器（RAM）或工作存储器中已有的块可以用新版本进行重写（再次装入它们）。原来的版本则被覆盖。

对 S7 块重新装入的步骤与下载相同。只是有一个注意出现，询问你是否要覆盖原来的块。

存在 EPROM 中的块不能被删除，但是一旦它被重新装入，原来的块就被声明无效了。替代它的块被装入 RAM 中。这就会在装载存储器或工作存储器中产生间隙。这些间隙最终意味着无法装入新的块，你该对存储器作压缩了。

注意

如果掉电又恢复而 RAM 没有备份电池，或者在 CPU 作了一个存储器复位之后，“旧”块又重新有效了。

17.1.6 通过 EPROM 存储卡下载

要求

用编程设备为 S7 可编程控制器访问 EPROM 存储卡必须有合适的 ERPOM 驱动器。为 M7 可编程控制系统访问 EPROM 存储卡，则必须安装了闪存文件系统（只能在 PG720、PG740 和 PG760 上）。当安装 STEP 7 标准软件包时，EPROM 驱动器和闪速文件系统作为可选项提供。如果使用 PC，要存储到 EPROM 存储卡需要一个外置 EPROM 写入装置，也可以在晚些时候再安装驱动器。要这样作，通过 Start>Simatic>STEP 7>Memory Card Parameter Assignment 调用相应的对话框，或者通过控制面板（双击图标“ Memory Card Parameter Assignment（存储卡参数赋值）”）。

保存在存储卡上

要将块或用户程序保存到存储卡可按如下进行：

1. 在编程设备的槽口中插入存储卡。
2. 打开“ Memory Card（存储卡）”窗口，可用以下方式：
   - 的“ Memory Card ”按钮。如果有必要可用菜单命令 View>Toolbar 激活工具栏。
   - 选择菜单命令 File>S7 Memery Card>Open。
3. 打开或激活显示你要存储的块的下列窗口之一：下列窗口都可以：
   - 项目窗口，“ 在线 ”视窗
   - 项目窗口，“ 离线 ”视窗
   - 库窗口
   - “ Accessible Nodes（可访问站）”窗口
4. 选择“ Blocks ”文件夹或单个块并将它们拷备到“ S7 Memory Card ”窗口。
5. 如果一个块已存在在存储卡中，则有错误信息显示。这种情况下，删除存储卡中的内容然后从步骤2开始重复。

17.1.7 分别下载硬件组态和连接组态

17.1.7.1 下载组态参数到可编程控制器

技巧

在下载之前，使用Station > Check Consistency菜单命令，可以确保在你的站组态中没有错误。然后，STEP 7可以检查可下载的系统数据是否能从现有组态中生成。 进行一致性检查时找到的任何错误都显示在一个窗口中。

下载条件

- 使用一根MPI电缆，将编程器（PG）连接至CPU的MPI接口。
- 在一个网络连接的系统中（编程器连接至一个子网）：所有子网中的模块都必须具有不同的网点地址，实际组态必须与你所创建的网络组态相匹配。
- 当前组态必须与实际站的架构相匹配。
- 只有符合条件并没有错误的组态才能被下载到站中。同样，只有生成的系统数据块（SDB）才能下载到模块中。
- 如果站的架构中包括使用任选软件包组态并赋值参数的模块：任选软件包必须被授权。

步骤

- 选择菜单命令PLC > Download To Module。STEP 7会通过对话框的方式引导你进行操作。

整个可编程控制器的组态可以下载到 CPU 中。CPU 的参数将立即激活。其它模块的参数在起动时被传送到模块中。

注意

有些组态，例如每个机架的组态，不能下载到站中。出于一致性考虑，STEP 7 总会将完整的组态下载到站中。

改变CPU下载时的操作模式

当你触发功能 PLC > Download 时，你可以根据对话框的引导，在编程器上执行以下操作：

- 切换CPU为STOP模式

  （如果模式选择开关设为RUN-P或至CPU的连接通过口令授权）

- 压缩存储器

  （如果没有足够的连续闲置存储器）

- 将CPU切回RUN模式

17.1.7.2 第一次下载网络组态

在第一次下载之前，连接至子网的模块还没有其组态的网点地址，但有一个缺省地址。为了使你的网络正确运行，子网中的每个网点都必须具有不同的网点地址。

- 通过CPU连接MPI子网

  CPU的缺省地址为“ 2 ”。但是，在一个子网中，该地址你只能使用一次，因此对于其它CPU，你必须改变缺省网点地址。

- 现场总线PROFIBUS和CP工业以态子网

  通过这些子网运行的站的CP必须进行组态，并指定一个网点地址。在你通过子网下载和通讯之前，你应通过站的MPI 指定地址（详见“ SIMATIC NET、NCM S7 for PROFIBUS ”和“ NCM for Industrial Ethernet ”手册）。

如果网点不是S7站...

如果一个网点不是 S7 站，必须使用工具赋值网络和网点属性或为此进行转换。例如，对于 DP 从站，其 PROFIBUS 地址必须使用开关设置。

必须确保这些设置与网络视图中的对象设置相匹配（PG/PC、其它站、S5 站）。

改变DP从站的PROFIBUS地址

连接一个 PROFIBUS 子网的 DP 从站也必须具有唯一的 PROFIBUS 地址。如果你想连接的 DP 从站支持功能“ Set_Slave_Add ”（例如 ET 200C），你可以使用 STEP 7 指定地址：

在 SIMATIC 管理器和组态硬件中，你可以使用菜单命令 PLC > Assign PROFIBUS Address，指定一个新的 PROFIBUS 地址。

注意：如果你对当前指定地址没有把握，你可以逐一连接 DP 从站至 PG/PC，并重新指定地址。

改变S7站的网点地址

要改变 S7 站的预定网点地址，可如下进行：

1. 组态站；在 “ General ” 选项卡（ “ Interface ” 下的 “ Properties ” 按钮）中设置连接模块的网点地址（例如 CPU）。
2. 切换模块为 STOP 模式，使用连接电缆连接编程器至模块上的接口。
3. 确定连接模块的预定网点地址（例如，使用 SIMATIC 管理器中的菜单命令 PLC > Display Accessible Nodes）。
4. 使用新的网点地址，将组态下载至可编程控制器（即所连模块）：
   - 在站的视图（组态硬件）中，使用菜单命令 PLC > Download。
   - 在网络视图（NetPro）中，选择你想下载的站，并选择菜单命令 PLC > Download > Selected Stations。输入 “ 旧 ” （仍有效）的预定地址。

17.1.7.3 下载网络组态到可编程控制器

要求

在此，我们应假设整个项目已被组态，意思是指：

- 已组态所有站
- 已生成所有子网，并设置其属性
- 已组态连接（如果需要的话）
- 已设置PG/PC接口，通过所连接的子网在PG/PC和可编程控制器之间可以进行通讯。
- 已进行组态一致性检查

只有组态无错误，即子网中网络连接的所有模块都具有唯一的网点地址，并且其实际组态与你所创建的网络组态相匹配，才能通过子网（PROFIBUS 或 MPI），将组态下载到可编程控制器中。

17.1.7.4 下载网络组态变化

要求

所有子网中网络连接的模块都必须具有唯一的网点地址，并且其实际组态必须与你已创建的网络组态相匹配。

如果你连接一个新站至子网，并且在子网中已有预设网点地址，你应根据“ 第一次下载 ”一节中所述进行。

如何下载？

在编译网络组态后（菜单命令 Network > Save and Compile）或 PLC > Download > ...后，NetPro 生成可以解释 SDB 中信息的模块的系统数据块（SDB）。SDB 中可以包括连接表、网点地址、子网属性、输入/输出地址以及模块参数。

根据你所选择的下载菜单命令，可以将不同的内容下载到不同的可编程控制系统中。

注意

只有使用选项命令 PLC > Download > Connections and Network Gateways，你才能在 CPU 处于 RUN-P 模式时下载。对于所有其它选项命令，CPU 必须切换至 STOP 模式。

菜单命令 PLC> Download >	下载什么？	下载到何处？
... 所选站	连接表、网点地址、子网属性、输入/输出地址以及所选站的模块参数	至所选站
... 所选站和所选伙伴站	连接表、网点地址、子网属性、输入/输出地址以及所选站的模块参数和所选站的连接伙伴	至所选站和该站的所有连接伙伴站
... 子网上的站	连接表、网点地址、子网属性、输入/输出地址和模块参数	逐一至所选子网的所有站
... 所选连接	所选连接（可能有多个选择）	至局域站和（对于双路连接）至相应的伙伴站
... 连接和网关	连接信息（有可能是一个空的连接表）和网关信息	至所连接的模块（可以处于 RUN-P 模式）

步骤

1. 连接编程器至你想装载的网点所连接的子网。
2. 打开 NetPro。
3. 选择你想下载的站或网络视图中的子网（以使用菜单命令 PLC > Download > Stations on Subnet）。
4. 选择上述选项之一，以使用菜单命令 PLC > Download。

其它信息

有关下载菜单命令的详细信息，见上下文帮助（选择菜单命令，并按 F1 键）。

17.1.7.5 下载全局数据组态

在编译程序运行时，全局数据表中的数据可以转化为系统数据。如果在编译过程中没有显示错误，你可以将系统数据发送到 CPU 中：

- 选择菜单命令PLC > Download。

17.2 从可编程控制器上载到 PG/PC

17.2.1 从可编程控制器上载到 PG/PC

当执行以下操作时，这个功能支持你：

- 保存来自可编程程序控制器中的信息（例如，为了维护目的）
- 快速组态和编辑一个站，如果在你开始组态前硬件部分是可用的

保存来自可编程程序控制器的信息

这个措施是必需的，举例来说，如果这个版本的离线项目数据不能或部分不能在 CPU 上运行。这种情况下，你至少可以恢复在线的可用的项目数据并上载它们到你的编程设备上。

快速组态

如果你组态完硬件并重新启动站后，从可编程序控制器中上载组态数据到你的编程设备，那么输入站组态就较容易。这提供给你站组态和每个模块的类型。然后你要做的全部事情就是以更多的细节来定义这些模块（定货号），并且对它们进行参数赋值。

下列信息可上载到编程设备上：

- S7-300：中央机架和任何扩展机架的组态
- S7-400：带一个CPU和信号模块的中央机架的组态，没有扩展机架
- 分布式I/O的组态数据不能上载到编程设备上

如果在可编程序控制器上没有组态信息，则这个信息将被上载；例如，如果在系统上执行了一个存储器的复位。否则，上载功能可提供更好的结果。

对于没有分布式 I/O 的 S7-300 系统，你要做的全部事情就是以更多的细节来定义这些模块（定化号），并且对它们进行参数赋值。

注意

当上载数据时（如果没有一个离线组态），STEP 7 不能确定全部组件的定货号。

当你用命令菜单 Options>Specify Module 组态硬件时，可以输入“未完成的”定货号。这样，你可以对 STEP 7 不能识别的模块进行参数赋值（那是指，没有出现在“硬件样本”窗口里的模块）；不管怎样，STEP 7 将不会检查你是否遵守参数法则。

从可编程序控制器上载时的约束

下列约束适用于从可编程序控制器上载到编程设备的数据：

- 块不包含任何参数、变量和标号的符号名称
- 块不包含任何注释
- 上载带有全部系统数据的完整的程序，只有这样系统才能继续处理属于“组态硬件”应用程序的系统数据
- 不能更进一步地处理全局数据通信（GD）和组态符号相关报文
- 强制作业不能把其他数据一起上载到编程设备上。它们必须作为变量表（VAT）被单独保存
- 模块对话框里的注释不能上载
- 只有在组态期间，这个选项被选中，模块的名称才会被显示（HWConfig：选项“在可编程序逻辑控制器中保存对象名称”在Options>Customize下的对话框里）

17.2.2 上载站参数

用命令菜单 PLC>Upload Station，可以从所选的可编程控制器中上载当前的组态、和所有块到编程设备上。

要作到这一步，STEP 7 在当前项目中生成一个新站，站的组态将存储在这个项目下。你可以修改这个新工作站的预定名称。（例如，“ SIMATIC 300-Station（1） ”）。这个插入的工作站在在线视图和离线视图中都会显示。

当一个项目打开时，这个命令菜单会被选中。是在这个项目窗口选择一个对象还是在视图（在线或离线）中选择一个对象对这个菜单命令没有影响。

- 你可以用这个功能更容易地组态
- 对于S7-300可编程控制器，被上载的实际硬件组态包括扩展机架，但是没有分布式I/O（DP）
- 对S7-400可编程序控制器，被上载的机架组态没有扩展机架和分布式I/O

对于没有分布 I/O 的 S7-300 系统，你要做的全部事情就是对模板作更详细的定义（定货号）并对它们进行参数赋值。

上载工作站时的限制

下列限制适用于数据上载到编程设备：

- 块不包含任何参数、变量和标号的符号名称。
- 块不包含任何注释。
- 包括所有的系统数据在内的完整的程序被上载，由此并非所有数据都能被进一步处理。
- 全局数据通信（GD），组态符号相关的报文和组态网络的数据不能被进一步处理。
- 强制作业不能被上载到编程设备上然后再装载回可编程序控制器上。

17.2.3 从一个 S7CPU 中上载块

你可以用 SIMATIC 管理程序，从 CPU 中上载 S7 块到编程设备的硬盘上。在下列情况中，上载块到编程设备上是有用的：

- 制作一个装载到CPU中的当前用户程序的备份。这个备份可以接着被重新下载，例如，维修人员维修后或CPU的一个存储器复位后。
- 你可以从CPU中上载用户程序到编程设备上并在那里编辑它，例如，为了故障检修的目的。这种情况下，你不能访问到程序文档的符号或注释。因此，我们建议这一过程仅应用于维修目的。

17.2.4 在 PG/PC 中编辑上载块

能从 CPU 中上载块到编程设备上有下列用途：

- 测试阶段中，可以在CPU上直接修改一个块，并为结果制作文档
- 可以通过装载功能从CPU的RAM装载存储器中上载块的当前内容到你的编程设备上

注意

在线和离线工作时，时间标志冲突。
以下过程导致时间标志冲突，因此要避免。

（如果按下列步骤做）当你在线打开一个块时，会有时间标志冲突：

- 在线所作的修改未保存到离线的S7用户程序中
- 离线所作的修改未下载到CPU

（如你按下列步骤做）当你离线打开一个块时，会有时间标志冲突：

- 一个在线块的时间标志冲突被复制到离线的S7用户程序中，并且在离线时被打开

两种不同情况

当从 CPU 上载块到编程设备时，记住有两种不同情况：

1. 块所属的用户程序在编程设备上。
2. 块所属的用户程序在编程设备上。

这表示下面列举的不能下载到 CPU 的程序部分就无法得到。这些内容是：

- 存有地址的符号名和注释的符号表
- 梯形逻辑或功能块图程序的段注释
- 语句表程序的行注释
- 用户定义的数据类型

17.2.5 重新下载硬件组态和连接组态

17.2.5.1 从一个站点上载组态参数

要求

你已经使用 MPI 表连接编程器（PG）至 CPU 的 MPI 接口。

技巧

上载站至一个新创建的空项目。

以特殊方式与其它相关的站（DP 主站上的 I 从站、使用直接数据交换/横向通讯链路中的接收器和发送器），应与项目一起上载。原因：如果没有该类站的特殊“ 伙伴 ”，项目会不一致。

步骤

1. 选择菜单命令 PLC>Upload。
2. 出现打开组态对话框。
3. 选择以后保存组态的项目并以“ OK ”确认。
4. 在随后出现的对话框中，设置网点地址、机架编号和读取组态（一般为 CPU）模块中的插槽。用“ OK ”确认。

你可以使用菜单命令 Station > Properties，赋值该组态的站名，然后将该站名保存在缺省项目中（菜单命令 Station > Save）。

17.2.5.2 上载网络组态

概述

你可以将项目的实际网络架构一个站一个站地上载到你的编程器中。

首先，你可以一个站一个站地将整个组态上载到 SIMATIC 管理器中的编程器中（菜单命令 PLC > Upload）。然后，STEP 7 在当前项目中为你想所上载的每个站创建一个新的站对象。

另外，当你组态硬件时，你也可以上载站组态（菜单命令 PLC > Upload）。

下面阐述如何将整个网络组态一个站一个站地上载到 NetPro 中。

要求

连接 PG/PC 至与你想上载的站或通过网关访问的站相同的子网中。已知连接子网的网点地址和模块的机架/插槽。

步骤

1. 连接编程器至你想装载的网点所连接的子网。
2. 如果需要，创建一个加载网络组态的新项目。
3. 通过以后你想保存上载网络组态的项目，打开 NetPro（例如通过一个新创建项目）。
4. 选择菜单命令 PLC > Upload Station。
   只有当一个项目打开时，才能选择这个菜单命令。
5. 在下面对话框中，通过给出其网点地址和机架/插槽，指定所要上载的站。

   在所有模块都网络连接的网络视图中，将出现“Station（站）”对象。另外还显示连接站的子网。使用菜单命令 Edit > Object Properties，你可以改变由系统指定的站名。

   当你选择一个连接终点的模块时，组态的连接也可以上载并显示。
6. 你可以修改站的组态或连接，然后将改变输入到站中。对于使用任选软件包创建的连接，必须安装任选软件包，以编辑这些连接，并输入到站中。
7. 如上所述进行操作，直至装载所有所需的站。
8. 如果需要的话，你可以将网络组态保存在当前项目中（菜单命令 Network > Save 或 Network > Save and Compile）。

上载编程器的连接特性

在连接表中，缺少离线组态的连接伙伴，即未指定连接伙伴。但是，根据属性对话框，在对话框中可以找到详细的地址。

PTP 连接的通讯方向在任何情况下都不能由 STEP 7 确定；但是，报文可以通知你哪个通讯方向合理。

17.2.6 在可编程序控制器中删除

17.2.6.1 清除装载/工作存储器并复位 CPU

在下载你的用户程序到 S7 可编程序控制器之前，你要在 CPU 上完成一个存储器复位，以保证没有旧块仍留在 CPU 上。

存储器复位的要求

CPU 必须在 STOP 模式下完成一个存储器的复位（模式选择开关设置为 STOP，或设成 RUN-P 并用菜单命令 PLC>Operating Mode 更改模式为 STOP）。

在一个S7 CPU上执行存储器的复位

当在 S7 CPU 上执行存储器复位时，有下列情况出现：

- 这个CPU复位
- 全部用户数据被删除（除了MPI参数以外的块和系统数据块（SDB））
- CPU中断全部现有的联接
- 如果数据保存在一个EPROM中（存储卡或集成EPPOM），随存储器复位后，CPU把EPROM的内容复制回存储器的RAM区。

诊断缓冲区的内容和 MPI 参数将被保留。

在M7 CPU/FM上执行存储器的复位

当在 M7CPU/FM 上执行存储器复位时，会有下列情况出现：

- 初始状态被恢复
- 除了MPI参数以外的系统数据块（SDB）被删除
- CPU/FM断开全部现有的联接。在你把CPU从STOP转换到RUN后，用户程序被保留并继续运行

使用“存储器复位”功能，当出现严重错误时，可通过删除工作存储器中当前系统数据块（SDB）并从只读存储器中重新装载 SDB 来恢复 M7CPU 或 FM 的初始状态。在有些情况下，要求操作系统作暖起动。要这么做，可使用模式选择开关（切换到 MRES 位置）清除 M7。只有当 CPU/FM 上使用的是 RMOS32 操作系统时才能够用模式选择开关在 SIMATIC M7CPU 或 FM 上作复位。

17.2.6.2 删除可编程序控制器上的 S7 块

在 CPU 程序的测试阶段，删除 CPU 上的单个块可能是必要的。块被保存在 CPU 用户存储器的 EPROM 中或 RAM 中（依据 CPU 和装载步骤）。

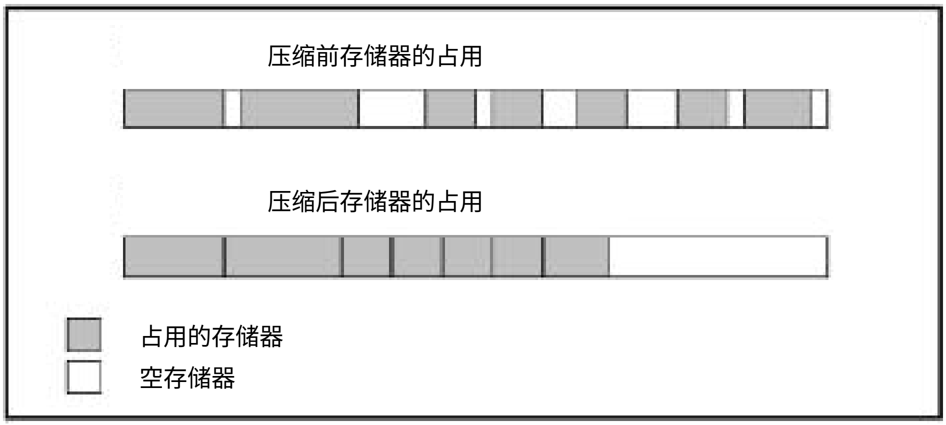
- RAM中的块可被直接删除。装载或工作存储器被占据的空间将空出来并可以被重新使用。
- 随着CPU存储器的复位，集成EPROM中的块总是被复制到RAM区。对于RAM中的备份可以直接作删除。被删除的块接着会在EPROM中被标记为无效，直到下一次存储器复位或没有RAM备份的电源掉。在存储器复位或没有备份RAM的情况下电源掉电后，“删除”的块从EPROM从中被复制到RAM中并且生效。集成EPROM中的块（例如，在CPU312中）将通过用新的RAM内容来重写它们而被删除。
- EPROM存储卡只能在编程设备上被擦除

17.2.7 压缩用户存储器（RAM）

17.2.7.1 用户存储器里的间隙（RAM）

删除和重装块之后，用户存储器（装载和工作存储器）将出现间隙，并且减少了可用的存储区。用压缩功能，现有的块将在用户存储器中无间隙地重新排列，一个连续的空存储器将产生。

下图显示了如何用压缩功能将占用存储器的块移到一起。



尽可能地在STOP模式下压缩存储器

有在 “ STOP ” 模式下压缩存储器才能够将所有间隙闭合。在 RUN-P 模式时（模式选择开关设置），当前正在处理的块由于被打开了而无法移动。在 RUN 模式（模式选择开关设置）（写保护!）下压缩功能不起作用。

17.2.7.2 压缩 S7 CPU 的存储器内容

压缩存储器的方法

有两种压缩用户存储器的方法如下：

- 当你向可编程控制器作下载时，如果没有足够的存储器可用，会出现一个对话框告之这个错误。你可以通过点击对话框中相应的按钮压缩存储器。
- 作为预防措施，你可以显示存储器的使用情况（用菜单命令PLC>Module Information，“ Memory（存储器） ”选项卡），如果需要的话可启动压缩功能。

步骤

1. 选择 “ Accessible Nodes ” 窗口或项目的在线窗口中的 S7 程序。
2. 选择菜单命令 PLC>Module Information。
3. 在随后出现的对话框中选择 “ Memory ” 选项卡。如果该 CPU 支持存储压缩功能的话，在这个选项卡中有一个用于存储器压缩的按钮。

18　用变量表进行测试

18.1　使用变量表测试简介

使用变量表，可以保存各种测试环境。这样，在操作或进行维修和维护时，可以有效地进行测试和监控。对于所保存的变量表的数量没有限制。

当使用变量表进行测试时，有如下功能：

- 监视变量

  该功能可以让你在编程设备/PC 上显示用户程序中或 CPU 中每个变量的当前值。

- 修改变量

  你可以用这个功能将固定值赋给用户程序或 CPU 中的每个变量。使用程序状态测试功能时也能立即进行一次数值修改。

- 使用外设输出并激活修改值

  这一功能允许你在停机状态下将固定值赋给 CPU 中的每个 I/O 输出。

- 强制变量

  你可以用这个功能给用户程序或 CPU 中的每个变量赋予一个固定值，这个值是不能被用户程序覆盖的。

你可以为以下变量赋值或显示数值：

- 输入、输出、位存储、定时器及计数器
- 数据块的内容
- I/O（外设）

在变量表中输入你想要显示或修改的变量。

你可以通过定义触发点和触发频率来决定变量什么时候、以什么样的频率被监视或赋予新值。

18.2 用变量表进行监视和修改的基本程序

要使用监视（Monitor）和修改（Modify）功能可按如下进行：

1. 生成新的变量表或打开已存在的变量表。
2. 编辑或检查变量表的内容。
3. 用菜单命令PLC>Connect to，建立当前变量表与所需的CPU之间的在线连接。
4. 用菜单命令Variable>Trigger，选择合适的触发点并设置触发频率。
5. 菜单命令Variable>Monitor和Variable>Modify，可使监视和修改功能在有效和无效之间转换。
6. 用菜单命令Table>Save或Table>Save As存储完成了的变量表，以便在以后的任何时候再调用它。

18.3 编辑并存储变量表

18.3.1 生成并打开一个变量表

在你能够监视或修改变量之前，必须生成一个变量表（VAT）并输入所需要的变量。要生成变量表可选择以下方法之一：

在SIMATIC管理器中：

- 选择“ Blocks ”（块）文件夹，使用菜单命令Insert>S7 Block>Variable Table。在对话框中，你可以给定表名（ “ Symbolic Name（符号）名 ” 文本框）。名称显示在这个对象窗口中。可以通过双击该对象打开变量表。
- 选择一个连接或者一个在线视窗、可访问站（accessible nodes）中的S7或M7程序。用菜单命令PLC>Monitor/Modify Variables，生成一个无名变量表。

在“ Monitor/Modify Variables（监视/修改变量）”中：

- 可以用菜单命令Table>New生成一个新的变量表，该表没有被赋给任何S7或M7程序。你可以用Table>Open打开已存在的表。
- 可以在工具栏中使用相应的符号来生成或打开变量表。

一旦已生成一个变量数，你可以存储、打印和多次使用该表进行监视和修改。

18.3.2 复制/移动变量表

你可以在一个 S7/M7 程序中的块文件夹中复制或移动变量表。

在复制或移动变量表时，应注意以下事项：

- 必须刷新目标程序符号表中的现有符号。
- 当移动变量表时，源程序符号表中的相应符号也将被移动到目标程序的符号表中。
- 当你从块文件夹中删除变量表时，S7/M7程序符号表中的相应符号也将被删除。
- 如果目标程序中包含一个具有相同名字的变量表，在复制变量表时，将被赋值更大的一个编号。
- 如果目标程序中已包含有一个相同名字的变量表，在复制时，你可以重新命名变量表（缺省为现有名字编号）。

18.3.3 存储变量表

当你再次测试程序时可以使用已存储的变量表来监视和修改变量。

1. 令Table>Save，存储变量表。
2. 如果已经生成变量表，现在你必须给变量表起个名字，例如“ ProgramTest_1 ”。

当你存储一个变量表时，所有当前设定和表的格式都被存储起来。这意味着在菜单选项“ Trigger ”下的设置也存储起来。

18.4 在变量表中输入变量

18.4.1 变量表中插入地址或符号

选择你想要修改或监视的变量，在变量表中输入。从“外部”开始，由“外”向“内”地工作，这意味着你首先应该选择输入，然后是被输入影响的变量以及影响输出的变量，最后是输出。

例如，如果你想监视输入位 1.0，存储字 5 以及输出字节 0，则在“Address”栏中输入以下内容：

例如：

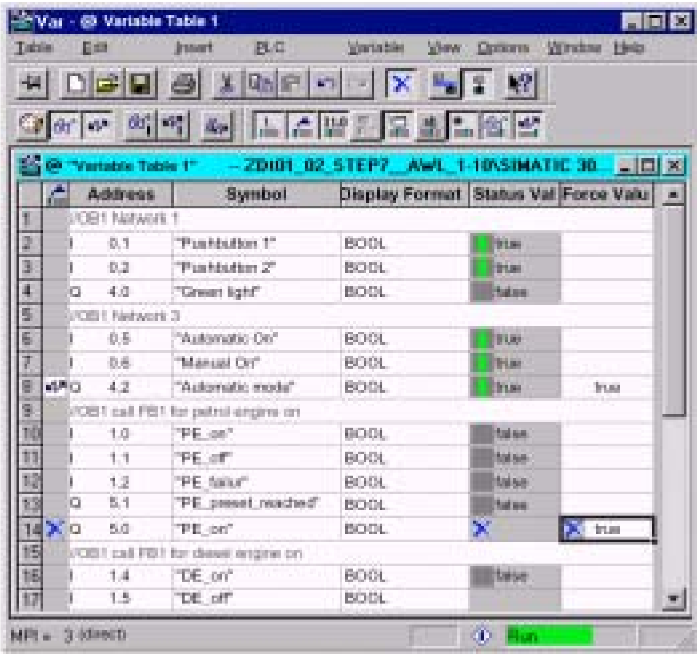
I 1.0

MW5

QB0

已完成的变量表的示例

下图所示是一个显示下述各栏的变量表：地址（Address）、符号（Symbol），显示格式（Monitor Format），监视值（Monitor Value）和修改值（Modify Value）



关于插入符号的注意

- 对你想要修改的变量输入地址或符号。即可以在“符号”栏输入符号，也可以在“地址”栏输入地址。然后输入便会自动地在正确的栏中写入。如果相应的符号已在符号表中定义，则符号栏或地址栏会自动填入。
- 你只能输入已在符号表中定义过的符号
- 符号必须准确地按照它在符号表中所定义的进行输入
- 符号名中含有特殊字符则必须用引号括起来（例如，“Motor.off”，“Motor+Off”，“Motor-off”）
- 要在符号表中定义新符号，可使用菜单命令Options>Symbol Table。还可以从符号表中拷贝符号然后粘贴到变量表中。

语法检查

当你在变量表中输入变量时，在每行的结束都会执行语法检查。任何不正确的输入都会被标为红色。如果你把光标放在红色的行上，可以从状态栏中读到错误的原因。按 F1 可得到关于错误纠正的注意。

最大限度

在变量表中每行最多可有 255 个字符。回车进入第二行是不可能的。一个变量表最多可有 1024 行。这是最大限度。

18.4.2 插入修改值

修改值作为注释

如果你想使变量的“修改值”无效，可以使用菜单命令 Variable > Modify Value as Comment。一个变量的修改值之前的注释符号“//”表示它已经无效。命令符号“//”也可以代替菜单命令调用，插在“修改值”的前面。通过再次调用菜单命令 Variable > Modify Value as Comment 或删除注释符号，可以取消“修改值”的无效性。

18.4.3 定时器输入的上限

注意，下面是输入定时器的上限：

例如：W#16#3999（BCD 格式的最大值）

例如

地址	监视格式	输入	修改值显示	解释
T 1	SIMATIC_TIME	137	S5TIME#130MS	转换为毫秒
MW4	SIMATIC_TIME	137	S5TIME#890MS	可用 BCD 格式表达
MW4	HEX	137	W#16#0089	可用 BCD 格式表达
MW6	HEX	157	W#16#009D	不能用 BCD 格式表达，所以监视格式不能选择为 SIMATIC_TIME

注意

- 你可以用毫秒级来输入定时器，但输入值会被作一些改变以适应时间结构。时间结构的大小要根据输入的时间值而定（137变成130ms，7ms被舍去）。
- 对于数据类型是WORD的地址的修改值，如IW1，被转换为BCD格式。但是，不是所有的位模式都是有效的BCD码。如果把一个数据类型是WORD的地址的输入表达为SIMATIC-TIME，则应用程序会自动转向缺省设置（这里是：HEX，请查看选择监视格式，缺省命令（“ view ”视窗菜单）），这样数值就可以被显示了。

用于SIMATIC-TIME格式的变量的BCD码

为变量类型为 SIMATIC_TIME 的变量输入数值时要用 BCD 码。16 个位具有以下含义：

| 0 0 x x | h h h h | t t t t | u u u u |

位15和14　　总是零

位13和12　　（标为xx）为位0至位11设置乘数：

00=>乘数为10毫秒

01=>乘数为100毫秒

10=>乘数为1秒

11=>乘数为10秒

位 11 至 8 为百位（hhhh）
位 7 至 4 为十位（tttt）
位 3 至 0 为个位（uuuu）

18.4.4 计数器输入的上限

注意下面计数器输入的上限：

计数器的上限是：C#999

W#16#0999（BCD 格式的最大值）

例如：

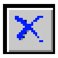
地址	监视格式	输入	修改值显示	解释
C1	COUNTER	137	C#137	转换
MW4	COUNTER	137	C#89	可用 BCD 格式表达
MW4	HEX	137	W#16#0089	可用 BCD 格式表达
MW6	HEX	157	W#16#009D	可表达为 BCD 格式，又不能选择 COUNTER 格式来监视

注意

- 如果你为计数器输入一个十进制数但却没有标识该值为C#，这个值会自动转为BCD格式（137变化C#137）。
- 对于数据类型是WORD的地址的修改值，如IW1，被转换为BCD格式。但是，不是所有的位模式都是有效的BCD码。如果，数据类型为WORD的地址的输入值无法表达为COUNTER，则应用程序会自动转向缺省格式（这里为：HEX，请查看选择监视格式，缺省命令（view视窗菜单）），这样数值就可以被显示了。

18.4.5 插入注释行

注释行以注释标志“ //　” 开始。

如果你想使一行或多行变量表无效（作为一个注释行），可以使用菜单命令 Edit > Line without effect 或工具栏中相应的符号。

18.5 举例

18.5.1 输入变量表的地址示例

允许的地址：	数据类型：	举例（英语助记符）：
Input | Output | Bit memory	BOOL	I 1.0 | Q 1.7 | M 10.1
Input | Output | Bit memory	BYTE	IB 1 | QB 10 | MB 100
Input | Output | Bit memory	WORD	IW 1 | QW 10 | MW 100
Input | Output | Bit memory	DWORD	ID 1 | QD 10 | MD 100
I/O （Input | Output）	BYTE	PIB 0 | PQB 1
I/O （Input | Output）	WORD	PIW 0 | PQW 1
I/O （Input | Output）	DWORD	PID 0 | PQD 1
Timers	TIMER	T 1
Counters	COUNTER	C 1
Data block	BOOL	DB1.DBX 1.0
Data block	BYTE	DB1.DBB 1
Data block	WORD	DB1.DBW 1
Data block	DWORD	DB1.DBD 1

注意：

“ DB0 ” 不允许使用，因为它已被内部占用。

在强制数值窗口：

- 对S7-300模板进行强制时，只有输入、输出和I/O（输出）是可以的。
- 对S7-400模板作强制时，只有输入、输出、位存储和I/O（输入/输出）是可以的。

18.5.2 输入连续的地址范围的示例

打开一个变量表并用菜单命令 Insert > Range of Variables 调用“ Insert Range of Variables（插入变量范围）”对话框。

作为对话框的输入，位存储的下列各行被插入到变量表中：

- From address（开始地址）：M 3.0
- Number（数量）： 10
- Display format（显示格式）：BIN

地址	显示格式
M 3.0	BIN
M 3.1	BIN
M 3.2	BIN
M 3.3	BIN
M 3.4	BIN
M 3.5	BIN
M 3.6	BIN
M 3.7	BIN
M 4.0	BIN
M 4.1	BIN

注意示例中“ 地址 ”栏中从第八项以后字节地址改变。

18.5.3 输入修改值和强制值的示例

位地址

可能的位地址	允许修改/强制值
I1.0	true
M1.7	false
Q10.7	0
DB1.DBX1.1	1
I1.1	2#0
M1.6	2#1

字节地址

可能的字节地址	允许修改/强制值
IB 1	2#00110011
MB 12	B#16#1F
MB 14	1F
QB 10	‘a’
DB1.DBB 1	10
PQB 2	-12

字地址

可能的字地址	允许修改/强制值
IW 1	2#0011001100110011
MW 12	w#16#ABCD
MW 14	ABCD
QW 10	B#(12,34)
DB1.DBW 1	‘ab’
PQW 2	-12345
MW3	12345
MW5	s5t#12s340ms
MW7	0.3s or 0,3s
MW9	c#123
MW11	d#1990-12-31

双字地址

可能的双字地址	允许修改/强制值
ID 1	2#00110011001100110011001100110011
MD 0	1.23e4
MD 4	1.2
QD 10	dw#16#abcdef10
QD 12	ABCDEF10
DB1.DBD 1	b#(12,34,56,78)
PQD 2	‘abcd’
MD 8	l#-12
MD 12	l#12
MD 16	-123456789
MD 20	123456789
MD 24	t#12s345ms
MD 28	tod#1:2:34.567
MD 32	p#e0.0

定时器

可能的”Timer”类型地址	允许修改/强制值	解释
T 1	0	转为毫秒
T 12	20	转为毫秒
T 14	12345	转为毫秒
T 16	s5t#12s340ms	
T 18	1.3	转为 1s300ms
T 20	1.3s	转为 1s300ms

修改定时器只影响其数值不影响状态。这意味着定时器 T1 的时间值可设为零，但却不会改变 A T1 的逻辑操作结果。

字符串 5t、s5time，大写或小写均可。

计数器

可能的 “ Counter ” 类型地址	允许修改/强制值
C 1	0
C 14	20
C 16	c#123

修改计数器只影响其数值不影响其状态。这意味着可将计数器 C1 的数值修改为零，却不会影响 A C1 的逻辑操作结果。

18.6 建立与 CPU 的连接

18.6.1 建立与 CPU 的连接

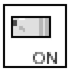
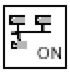
为了监视或修改在当前变量表（VAT）中输入的变量，你必须与相应的 CPU 建立连接。可以将每个变量表与不同的 CPU 建立链接。

显示在线连接

如果有在线连接存在，变量表窗口标题栏中的 “ ONLINE（在线） ” 项会显示该情况。状态栏将显示操作状态 “ RUN ”、 “ STOP ”、 “ DISCONNECTED ” 或 “ CONNECTED ”，这取决于 CPU。

建立与CPU在线连接

如果与所要求的 CPU 的在线连接不存在，使用菜单命令 PLC>Connect To>…来定义与所需的 CPU 的连接，以便进行变量的监视或修改。另外，还可以点击工具栏中相应的按键。

	与组态的 CPU 的连接
	与直接相连的 CPU 的连接（如，MPI=2（direct））

中断与CPU的在线连接

使用菜单命令 PLC>Disconnect，可以中断变量表和 CPU 的连接。

注意

如果用菜单命令 Table>New，生成了一个未命名的变量表，你可将它连接到最后组态的 CPU 上。

18.7 监视变量

18.7.1 监测变量

以下方法可用监视变量：

- 用菜单命令Variable>Monitor，激活监视功能。所选变量的数值依据所设定的触发点和触发频率显示在变量表中。如果设置触发频率为“ Every Cycle（每一循环）”，你可以用菜单命令Variable>Monitor，将监视功能切换成无效。
- 你可以使用用菜单命令Variable>Update Monitor Values，对所选变量的数值作一次立即刷新。所选变量的当前数值则显示在变量表中。

用ESC键放弃“ 监视 ”

如果在监视功能激活的状态下按 ESC 键，该功能则不经询问就退出。

18.7.2 为变量表定义触发

你可以在编程设备上显示用户程序中每个变量在程序处理过程中的某一特定点（触发点）的当前数值，以便对它位进行监视。

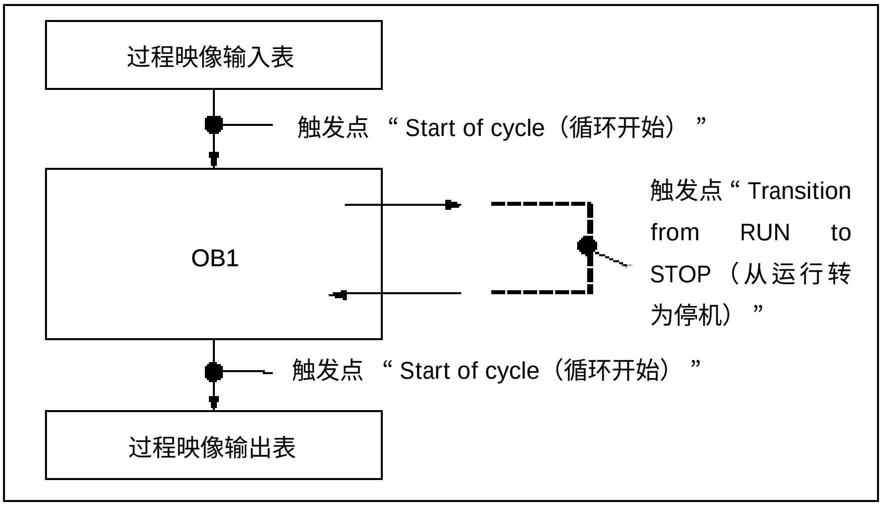
当你选中一个触发点时，就决定了监视的变量在哪个时间点的数值被显示出来。

可以用菜单命令 Variable>Trigger，设置触发点和触发频率。

触发	可能的设置
触发点	循环开始 循环结束 从RUN转换到STOP
触发频率	一次 每个周期

触发点

下图所示为触发点的位置。



要想监视修改值，你应将监视触发点设在 “ Status Value ” 栏上，将修改触发点设在 “ End of cycle ” 。

立即触发

你可以用菜单命令 Variable>Update Monitor Values，刷新所选变量的数值。这个命令即为 “ 立即触发 ”，不参照用户程中的任何点，尽可能快地执行。这些功能主要用于停机模式下的监视和修改。

触发频率

	触发频率：一次	触发频率：每一个循环
监视变量	依据触发点的不同只触发一次	在定义的触发点上监视 当测试一个块时，你可以准确地追踪处理的过程

18.8 修改变量

18.8.1 修正变量

你可以利用下述方法进行变量修改：

- 用菜单命令Variable>Modify，激活修改功能。在变量表中，根据触发点和触发频率的设定而对所选变量作的数值修改将应用到用户程序中。如果设置触发频率为“ Every Cycle（每一循环） ”，你可以用菜单命令Variable > Monitor，将监视功能切换成无效。
- 你可以用菜单命令Variable>Activate Modify Values，对所选变量的数据作一次立即刷新。

强制功能和外设输出（PQ）使能提供一些其它的可能性。

当进行修改时，注意：

- 只能对那些起动修改功能时在变量表中可以看到的地址进行修改。

  一旦你起动修改功能时缩小了变量表的可视区域，有一些也许已修改了的值却看不到了。如果把变量表可视区域变大一些，则可看到一些未修改的地址。
- 修改功能已经做了，就不能再取消（比如，用Edit>Undo）。
- 如果你选择了在每一循环进行修改，则无法滚动屏幕。

危险

在程序运行中修改变量值，如果功能或程序中出错可能导致对人身或财产的损害。在执行“ 修改 ”功能前，要确认不会有危险情况出现。

用ESC键放弃“ 修改 ”

如果在“ Modifying（修改） ”功能进行过程中你按了 ESC 键，该功能无询问退出。

18.8.2 为变量表定义触发

你可以在程序运行中的某一个特定点（触发点）给用户程序的每个变量赋予固定值（一次或每一循环）。

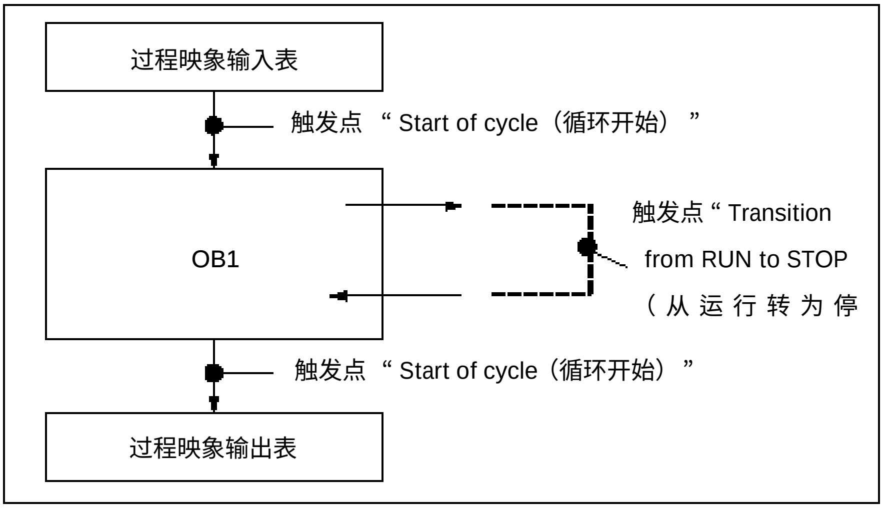
当你选中一个触发点时，就决定了监视的变量在哪个时间点的数值被显示出来。

可以用菜单命令 Variable>Trigger，设置触发点和触发频率。

触发	可能的设置
触发点	循环开始 循环结束 从运行转为停机
触发频率	一次 每一个循环

触发点

下图所示为触发点的位置。



所示为触发点的位置：

- 修改输入只适用于触发点 “ Start of cycle ” （对应于用户程序OB1的开始），因为在修改后将刷新过程映象。
- 修改输出只适用于触发点 “ End of cycle ” （对应于用户程序OB1的结束），因为用户程序可以覆盖输出的过程映象。

要想监视修改值，你应将监视触发点设在“ Status Value ”段上，将修改触发点设在“ End of cycle ”。

以下是变量修改时不同的触发点的情形：

- 如果你设置触发频率为“ Once ”，而所选变量不能被修改时，会显示信息。
- 对触发频率“ Every cycle ”则无信息出现。

立即触发

你可以用菜单命令 Variable>Activate Modify Values，对所选变量的数值进行修改。这个命令即为“ 立即触发 ”，不参照用户程中的任何点，尽可能快地执行。这个功能主要用于在停机模式下修改。

触发频率

下表所示为触发条件的设定对变量修改的影响：

	触发频率：一次	触发频率：每一个循环
修改变量	激活一次 与触发点无关，你可以将数值赋给变量一次	在定义的触发点进行修改 通过赋予固定数值，你可以为用户程序模拟某一情形，用这个功能可调试你已编好的功能

18.9 强制变量

18.9.1 强制变量

你可以给用户程序的每个变量赋予固定值，这样它们就不能够被 CPU 中正在执行的用户程序改变或覆盖。实现这一功能的要求是 CPU 支持该功能（如，S7-400 的 CPU）。通过将固定值赋给变量这一功能，你可以为用户程序设置特定的情形并用该方法对已编程的功能进行测试。

“强制数值（ForceValues）”窗口

只有当“强制数值”窗口处于激活状态，才能选择用于强制的菜单命令。

要显示这个窗口可选择菜单命令 Variable > Display Force Values。

你只需为 CPU 打开一个“强制值”窗口。激活的强制作业的变量和它们各自的强制值就都一起显示在窗口中。

强制数值窗口示例



当前在线连接的名字显示在标题栏中。

状态栏中显示的是从 CPU 中读出的强制作业的日期和时间。

如果没有激活的强制作业，该窗口是空的。

在“强制数值”窗口中不同的显示变量的方法有以下含义：

显示	含　义
黑体	该变量在 CPU 中已被赋予固定值
正常	该变量正在被编辑
灰色	模板的变量在机架上不存在/未插入或者变量地址错误，显示错误信息

使用变量表中的可强制地址

选择你要强制的变量表。

当你打开“Force Values”窗口时，如果模块可以强制变量，就可以使用这些变量。

使用CPU中的强制作业或建立新的强制作业

如果“强制数值”窗口是打开并且激活的，则有另外的信息显示：

- 如果你确认它，该窗口中的改变将会被CPU中已存在的强制作业覆盖。用菜单命令Edit>Undo，可以重新存储前一窗口的内容。
- 如果你取消它，当前窗口中的内容就保留下来。

然后你可以用菜单命令 Table>Save As，将“强制数值”窗口的内容存为一个变量表，或者选择菜单命令 Variable>Force：这样就将当前窗口的内容写到 CPU 中作为一个新的强制作业。

变量的监视和修改只能在变量表中进行，不能在“强制数值”窗口。

存储强制数值窗口

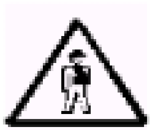
你可以将强制数值窗口中的内容存到一个变量表中。使用菜单命令 Insert>Variable Table，可以在一个强制数值窗口中重新插入已存储的内容。

在强制数值窗口中关于符号的提示

除非你从其它的没有符号的应用程序中打开“监视和修改变量”应用功能，否则最后激活的窗口中的符号会被输入。

如果你无法输入符号名，则“Symbol（符号）”这一栏被隐藏了。这种情况下，菜单命令 Option>Symlol Table 没有激活。

18.9.2 强制变量时的安全措施



注意人员伤害和财产损失

注意，当使用“强制”功能时，任何不正确的操作都可能会：

- 危及人员的生命或健康或者
- 造成设备或整个工厂的损失





小心（Caution）

- 在你开始强制功能之前必须检查确保同一时间在同一CPU上没有其他人在执行该功能。
- 一个强制作业只能用菜单命令Variable>Stop Forcing，来删除或终止。关闭强制数值窗口或退出“监视和修改变量”应用程序并不能删除强制作业。
- 强制功能不能被取消（例如，用Edit>Undo）。
- 请阅读有关信息，了解强制和修改之间的差别。
- 如果一个CPU不支持强制功能，则在变量菜单中与强制有关的所有菜单命令都不能激活。

如果用菜单命令 Variable>Enable Peripheral Output，使输出禁用失效，所有被强制的输出模板输出它们的强制值。

18.9.3 强制和修改变量的区别

下表总结了强制和修改的区别：

特点/功能	用 S7-400 强制	用 S7-300 强制	修改
位存储(M)	•	-	•
定时器和计数器(T,C)	-	-	•
数据块(DB)	-	-	•
外设输入(PIB,PIW,PID)	•	-	-
外设输出(PQB,PQW,PQD)	•	-	•
输入和输出(I,Q)	•	•	•
设置触发	总是立即触发	总是立即触发	一次或每个循环
功能只影响激活窗口中可视区域的变量	影响所有强制值	影响所有强制值	•
用户程序可覆盖修改/强制值	-	•	•
无中断地有效替换强制数值	•	•	-
当应用程序退出时变量仍保持其数值	•	•	-
与 CPU 的连接断开后变量仍保持其数据	•	•	-
允许的寻址错误： 如：IW1 修改/强制值：1 IW1 修改/强制值：0	-	-	最后的有效

注意

- “ 使能外设输出 ” 时，在相应输出模板上的强制外设输出的强制数值有效；而外设输出的修改值却无效。
- 使用强制功能，变量总是有被强制的值。这些值在对用户程序的读取访问过程中被读取。所有形式的写访问都无效。
- 用永久修改功能时，对程序的读访问是有效的并可保持到下一触发点。

19 用程序状态（Program Status）功能进行测试

19.1 用程序状态功能进行测试

你可以通过显示程序状态（RLO（操作结果），状态位）或为每条指令显示相应寄存器内容的方法测试你的程序。可以在“ Customize ”对话框中定义在“ LAD/FBD ”画面中的显示范围。在“ LAD/STL/FBD：Programming Blocks ”窗口中，使用菜单命令 Options > Customize 打开这个对话框。

警告

在过程运行过程中测试程序，如功能或程序出错，会对人员或财产造成严重损害。

在你开始这项功能之前，确认不会有危险情况出现。

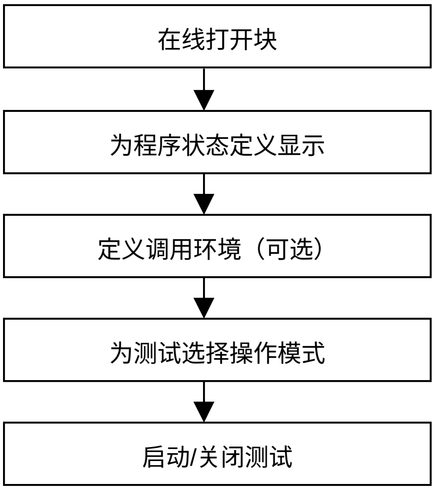
要求

要显示程序状态，必须满足下列要求：

- 你必须存储了没有错误的程序，并且将它们下载到CPU。
- CPU在运行并且用户程序在执行。
- 块必须在线打开。

监视程序状态的基本程序

建议不要调用整个程序进行调试，而应一个块一个块地调用并单独地调试它们。你应该从调用分层嵌套中最外层的块开始，例如，在 OB1 中调用它们，通过监视和修改变量功能为块生成被测试的环境。



程序状态中测试，并在单步模式下执行程序，测试的操作模式必须设置（见菜单命令 Debug＞Operation）。这些测试功能在过程操作模式下是不可能的。

19.2 程序状态显示

程序状态的显示是循环刷新的。它从所选网络开始。

LAD和FBD中的预设颜色代码

- 状态满足：绿色连续线
- 状态不满足：兰色点线
- 状态未知：黑色连续线

在 “ LAD/FBD ” 选项卡中，使用菜单命令 Options＞Customize，可改变线型及颜色的预置。

元素的状态

- 触点的状态是：
  - 如果该地址有“ 1 ”值则满足，
  - 如果该地址有“ 0 ”值则不满足，
  - 如果该地址的值不知道则为未知。
- 带有使能输出（ENO）的元素的状态相应于以ENO输出值为地址的触点的状态。
- 带有Q输出的元素的状态相应于有该地址值的触点状态。
- 如果BR位在调用功能后被置位，则调用（CALL）的状态满足。
- 如果跳转被执行则跳转指令的状态满足，即意味着跳转条件满足。
- 带有使能输出（ENO）的元素，如果使能输出未被连接则该元素显示为黑色。

线的状态

- 线的状态如果未知或没有完全运行则是黑色的。
- 在电力总线开始处的线的状态总是满足的（“ 1 ”）。
- 并行分支开始处线的状态总是满足的（“ 1 ”）。
- 如果一个元素和它前面的线的状态都满足则该元素后面的线的状态满足。
- 如果NOT指令前面的线的状态不满足（相反）则NOT指令后面的线状态满足。
- 在许多线的与逻辑后面的线的状态可以满足，如果：
  - 与之前至少有一条线的状态被满足。
  - 分支前的线的状态满足。

参数状态

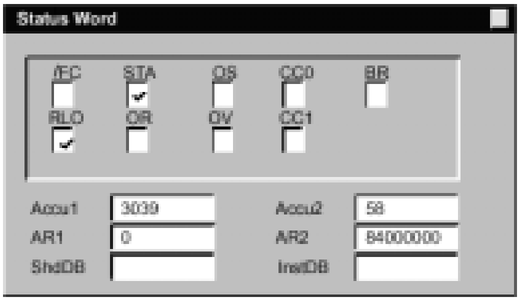
- 黑体类型的参数值是当前值。
- 细体类型的参数值来自前一个循环；该程序区在当前扫描循环中未被处理。

19.3 你应了解的关于在单步模式/断点下进行测试的信息

在单步模式下进行测试，你可以做以下事情：

- 一条指令一条指令地执行程序（在单步模式下）
- 设置断点

“ 在单步模式下测试 ” 的功能并不是在所有的可编程控制器上都是可能的（参考相关的可编程控制器的资料）。



要求

- 必须对测试模式进行设置。在过程操作模式下单步测试模式是不可能的（见菜单命令Debug＞Operation）。
- 用单步执行模式测试只有在语句列表的情况下才是可能的。对于在梯形逻辑或功能块图方式下的块，你必须用菜单命令View > STL改变视窗。
- 该块不能是被保护的。
- 块必须在线打开。
- 已打开的块不能在编辑器中被修改。

断点的数量

断点的数量依据下列情况而变化：

- 已设置的断点数
- 运行的变量状态的数目
- 运行的程序状态的数目

参考你的可编程控制器的资料，查看它是否支持单步执行模式。

在“ Debug（调试） ”菜单中，你可以找到菜单命令用来设置、激活或删除断点。还可以用断点栏中的图标选择这些菜单命令，用菜单命令 View＞Breakpoint Bar 显示断点栏。

允许的测试功能

- 监视/修改变量
- 模板信息
- 操作模式

危险（Danger）

在 HOLD（保持）模式下危险系统状态

19.4 你应该了解的关于 HOLD（保持）模式的信息

如果程序遇到断点，可编程控制器进入 HOLD 操作模式。

HOLD模式下LED的显示

- LED RUN 闪烁
- LED STOP 亮

在HOLD模式下程序的处理

- 在HOLD模式下不处理S7指令码，这意味着没有优先级被进一步处理。
- 所有定时器被冻结：
  - 没有定时器单元被处理
  - 所有监视时间停止
  - 时间控制层次的基本时钟速率被停止
- 实时时钟继续运行
- 为安全原因，在HOLD模式下输出总是禁止的（“ 输出禁止 ”）。

HOLD模式下电源掉电后的性能

- 有备份电池的可编程控制器在HOLD模式下，随着电源掉电随后又恢复供电转为STOP模式并保持在这儿。CPU不执行自动再启动。在STOP模式下你可以决定过程如何继续（例如，设置/清除断点，执行一个手动再启动）。
- 没有备份电池的可编程控制器不具有“ 记忆 ”，所以当电源恢复后执行一个自动暖启动，不考虑以前的操作模式。

19.5 编程数据块状态

版本 5 以上的 STEP 7 都可以在数据视窗中在线观察数据块。显示屏可以由在线数据块也可以由离线数据块激活。在这两种情况下，将显示可编程控制器中在线数据块内容。

在启动程序状态之前，不能修改数据块。如果在线数据和离线数据块之间有结构差异（声明），可以根据需要将离线数据块直接下载到可编程控制器中。

数据块必须位于“ data view（数据视窗）”中，以使在线数值可以显示在“ Actual Value（实际数值） ”栏中。只有显示屏上可以看见的数据块部分才能刷新。在状态激活时，你不能切换为声明视窗。

在正在刷新时，在状态栏中将显示一个绿框，并显示操作模式。

数值分别以各自的数据类型显示；其格式不能修改。

在结束程序状态后，“ Actual Value（实际数值） ”栏将显示程序状态之前的有效内容。不能将刷新的在线数值传送至离线数据块。

刷新数据类型：

在共享 DB 以及实例数据块的所有声明（in/out/inout/stat）中，都可以刷新所有基本数据类型。

有些数据类型不能刷新。当程序状态激活时，包含还没有刷新数据的‘ Actual Value（实际数值） ”栏中的区域将显示为灰色背景。

- 复合数据类型DATE_AND_TIME和STRING不能刷新。
- 在复合数据类型ARRAY、STRUCT、UDT、FB和SFB中，只有那些基本数据类型元素才能刷新。
- 在一个实例数据块中的INOUT声明中，只显示复合数据类型指示符，不显示数据类型的元素本身。指示符不能刷新。
- 参数类型不能刷新

19.6 为块设置调用环境

要记录程序状态，可以通过设置调用环境指定确切的条件。于是程序状态只在满足触发条件时才被记录。

要设置显示可按如下进行：

1. 单命令 Debug>Call Environment。
2. 框中设置触发条件并用 “ OK ” 确认它们。

选　项	含　义
调用路径	这里可以指定调用路径，在该路径上被测块一定被调用以激活状态记录。你可以输入到达被测块前最近的三个调用层
有地址	如果调用路径条件应去活，应取消。
打开数据块	这里通过指定一个或两个数据块来指定调用环境。如果被测块调用了指定的数据块，则记录其状态。

为块背景指定调用环境

为了使一个块的程序状态显示在一个特定的背景中，应如下操作：

1. 单命令 Debug > Mode，并设置 “ Debug mode（调试模式） ” 操作模式
2. 调用的块，并将光标放在所要调用指令上（STL 中的 CALL 行或 LAD/FBD 中的块中的框）。
3. 右键，使用 Call Path，选择菜单命令 Called Block > Monitor。

   结果：打开调用的块。调用作为块的触发条件输入，并激活块的该背景状态。

数据块的现有触发条件不改变。

20 使用模拟程序 S7- PLCSIM（可选软件包）进行测试

20.1 使用模拟程序（可选软件包）进行测试

有可选软件包 PLC Simulation（模拟 PLC），你可以在计算机或编程设备（如 PG740）上的模拟可编程控制器上运行并测试你的程序。由于该模拟功能可完全由 STEP 7 软件实现，你不需要任何 S7 硬件（CPU 或信号模板）。使用模拟的 S7 CPU，你可以对 S7-300 和 S7-400 CPU 作程序测试和故障诊断。

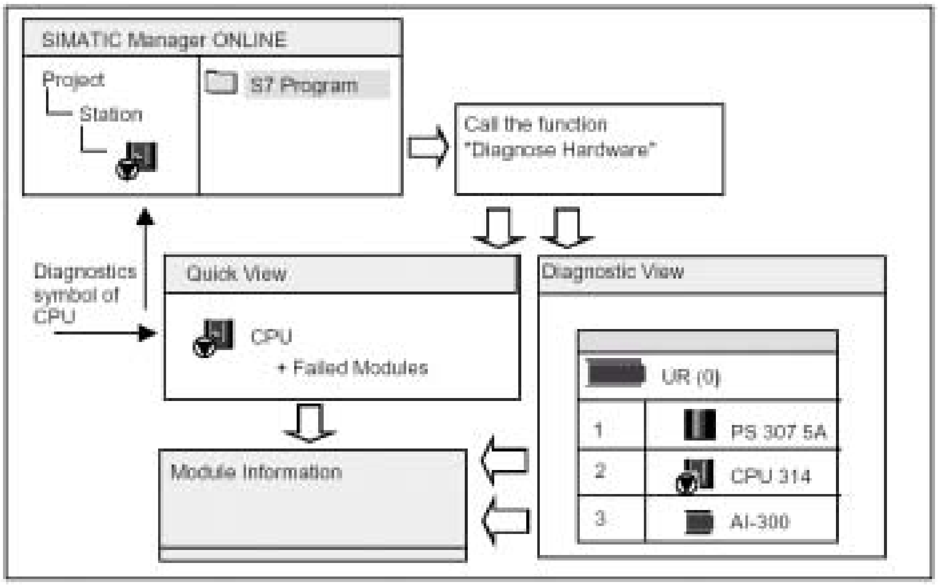
这个应用程序为监视和修改各种用户程序中使用的参数提供了一个简单的用户界面（如，输入开关接通和断开）。程序由模拟的 CPU 处理时，你还可以在 STEP 7 中使用各种应用程序。例如，可以用变量表监视和修改变量。

21 诊断

21.1 诊断硬件和故障诊断

你可以通过观察诊断符号来判断一个模板是否有诊断信息。诊断符号可显示相应模板的状态，对于 CPU，还可显示操作模式。

诊断符号可显示在在线项目窗口以及快速视窗（缺省设置）或调用“ Diagnose Hardware（诊断硬件）”功能时的诊断视窗中。更详细的诊断信息显示在“ Module Information（模板信息） ” 应用程序中，你可以通过双击快速视窗或诊断视窗中的诊断符号起动该应用程序。



怎样进行故障定位

1. 用菜单命令 View＞Online，打开项目的在线窗口。
2. 打开所有的站，以便看到其中组态的可编程模板。
3 查看哪个 CPU 显示了指示错误或故障的诊断符号。可以用 F1 键打开对诊断符号进行解释的帮助页。

4　　选择你想要检查的站。

5.　选择菜单命令 PLC＞Module Information，为该站中的 CPU 显示模板信息。

6.　选择菜单命令 PLC＞Diagnose Hardware，显示 CPU 的“ quick view ”（快速视窗）”及本站中的故障模板。快速视窗的显示被设作缺省设置（菜单命令 Option＞Customize，“ View ”选项卡）。

7.　在快速视察中选择故障模板。

8.　点击“ Module Information（模板信息）”按键，以获得该模板的信息。

9.　点击快速视窗中的“ Open Station Online ”按键，显示诊断视窗。诊断视窗中包含了该站中按槽口顺序排列的所有模板。

10.　双击诊断视窗中的模板以显示其模板信息。用这种方式，还可以得到那些没有故障因而没有显示在快速视窗中的模板信息。

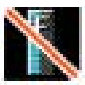
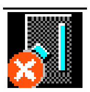
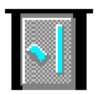
你不一定需要执行上述所有各步，当你得到所需的诊断信息后就可以停止。

21.2　在线视窗中的诊断符号

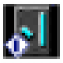
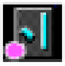
诊断符号显示在在线项目窗口和在线组态表的硬件组态窗口。

诊断符号使得你的故障检查更容易。你只要看一眼模板符号，就知道诊断信息是否已经有了。如果没有故障出现，模板类型的符号显示则没有附加的诊断符号。

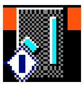
如果模板的诊断信息已经有了，则会有在模板的符号上增加一个诊断符号，或显示的模板符号对比度降低。

符　号	模　式
	当前组态与实际组态不匹配（module exists/type monitoring mismatch）：被组态的模板不存在，或者插入了不同类型的模板
	故障：模板出现故障 可能的原因：诊断中断，I/O 访问错误，或检查到故障 LED
	无诊断可能。原因：无在线连接或该 CPU 不支持模板诊断信息（如：低压电源或分模数）

模板的诊断符号（举例：FM / CPU）

符　号	模　式
	起动（STARTUP）
	停机（STOP）
	停机（STOP）在多机操作模式下由另一个 CPU 触发的停机
	运行（RUN）
	保持（HOLD）

操作模式的诊断符号（举例：CPU）

符　号	含　义
	在该模板上有变量被强制，即在模板的用户程序中有变量被赋予一个固定值，该数据值不能被程序改变。 强制符号还可以与其它符号组合在一起显示（这里是与运行（RUN）模式符号一起）

强制的诊断符号

刷新诊断符号的显示

激活相应的窗口。

- 按F5或
- 在窗口中选择菜单命令View＞Update。

21.3 诊断硬件：快速视窗

21.3.1 访问快速视窗功能

快速视窗提供一种使用“ Diagnosing Hardware（硬件诊断） ”的快速方法，它所显示的信息比硬件组态中诊断视窗所显示的细节要少。当“ Diagnos Hardware ”功能调用时，快速视窗作为缺省设置显示出来。

显示快速视窗

在 SIMATIC 管理器中用菜单命令 PLC＞Diagnose Hardware，调用该功能。

你可以在下列情况下使用菜单命令：

- 如果在一个在线项目窗口中选中了一个模板或一个S7/M7程序。
- 如果在“ Accessible Nodes（可访问网点）”窗口中选中了一个站（“ MPI=… ”）并且该选项属于一个CPU。

从显示的组态表中，你可以选择想要显示其模板信息的模板。

21.3.2 快速视窗中的信息功能

下列信息会显示在快速视窗中：

- 在线连接的CPU的数据
- CPU的诊断符号
- 已查到故障的CPU的模板诊断符号（例如，诊断中断、I/O访问错误）
- 模板类型和地址（机架、槽、带有站号的DP主系统）

快速视窗中的其它诊断选项

- 显示模板信息

  你可能通过点击“ Module Information（模板信息） ”键，调用该对话框。对话框依据所选模板的诊断能力显示详细的诊断信息。特别地，你可以通过 CPU 的诊断信息显示诊断缓冲区中的各项内容。

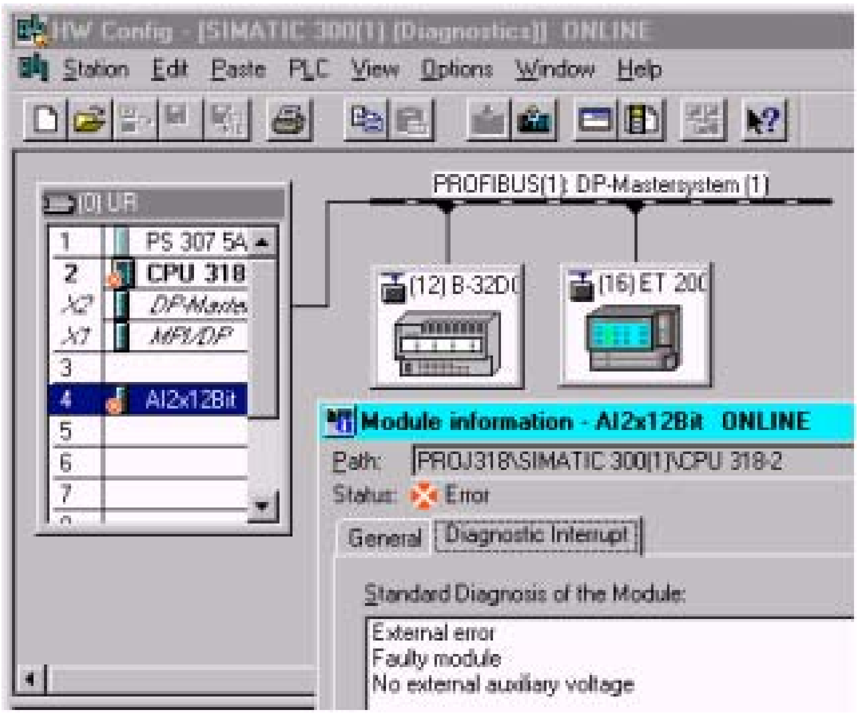
- 显示诊断视窗

  使用“ Open Station Online（打开在线站）”按键，可以打开一个对话框，与快速视窗相比，该对话框中包含整个站的图形概述以及组态信息。它集中在“ CPU/Faulty Modules（CPU/故障模板）”列表中被切亮的模板上。

21.4 诊断硬件：诊断视窗

21.4.1 调用诊断视窗

使用这种方法你可以为机架上的所有模板打开“ Modufe Information ”对话框。诊断视窗（组态表）显示不同层的机架中站的实际结构，以及 DP 站及其模板。



注意

- 如果离线的组态表已经打开，你可以使用菜单命令Station＞Open Online，打开组态表的在线视窗。
- 根据模板的诊断能力的不同，在“ Module Information ”对话框中显示不同数量的选项卡。
- 在“ Accessible Nodes ”窗口，只有那些有自己的站地址（MPI或PROFIBUS地址）的模板才能够看得见。

从SIMATIC管理器的在线项目视窗中调用

1. 在 SIMATIC 管理器中，使用菜单命令 View＞Online，与可编程控制器建立在线连接。
2. 选择一个站并双击打开。
3. 然后打开其中的“ Hardware ”对象。诊断视窗就打开了。

现在你可以选择一个模板，使用菜单命令 PLC＞Module Information 调用其模板信息。

从SIMATIC管理器的离线项目视窗中调用

执行以下步骤：

1. 从 SIMATIC 管理器的项目视窗中选择一个站并双击打开。
2. 然后打开其中的“ Hardware ”对象。组态表就打开了。
3. 选择菜单命令 Station＞Open Online。
4. 由模板（如，CPU）所决定的站组态及硬件组态诊断视窗就打开了。模板的状态由符号来指示。各种符号的含义可参考在线帮助。故障模板以及已经组态但找不到的模板分别列在不同的对话框中。在这个对话框中你可以直接去找选中的模板（“ Go To ”键）。
5. 双击你感兴趣的模板的符号。带有选项卡（依模板类型而不同）的对话框可以给你有关模板状态的详细分析。

从SIMATIC管理器的“ Accessible Nodes ”窗口中调用

执行以下步骤：

1. 使用菜单命令 PLC＞Display Accessible Nodes，在 SIMATIC 管理器中打开 “ Accessible Nodes ”窗口。
2. 在“ Accessible Node ”窗口选择一个站。
3. 选择菜单命令 PLC＞Diagnose Hardware。

注意

在“ Accessible Nodes ”窗口，只有那些有自己的站地址的模板才能够看得见。

21.4.2 诊断视窗中的信息功能

与快速视窗相比，诊断视窗显示在线已有的整个站的组态。包括：

- 机架组态
- 所有组态模板的诊断符号
  从这里你可以读每个模板的状态以及CPU模板的操作模式。
- 模板类型、序号和地址细节、有关组态的注释。

诊断视窗中另外的诊断选项

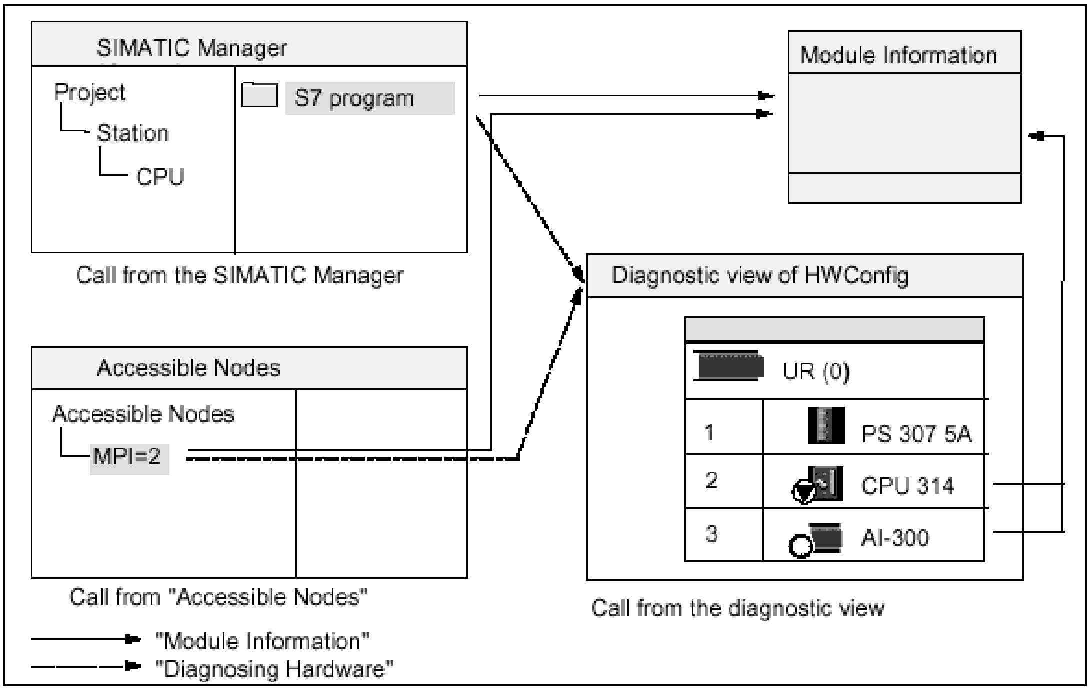
双击一个模板，你可以显示该模板的操作模式。

21.5 模板信息

21.5.1 显示模板信息的选项

你可以从不同的起点显示“ Module Information（模板信息） ”对话框。下面的程序是常用的调用模板信息的方法举例：

- 在SIMATIC管理器中从项目的“ 在线 ”或“ 离线 ”视窗。
- 在SIMATIC管理器的“ Accessible Nodes（可访问站） ”窗口中。
- 在硬件组态的诊断视窗。



为了要显示有自己的站地址的模板状态，你需要可编程控制器的在线连接。你可以通过项目的在线视窗或通过“ 可访问站 ”窗口建立这种连接。

21.5.2 模板信息功能

在“ Module Information ”对话框中，不同的选项卡中有不同的模板信息功能。在激活状态下显示时，只有那些与所选模板相关的选项卡显示出来。

功能/选项卡	信　息	用　途
概述	所选模板的标识数据；比如，定货号、发行号、状态、机架上的槽	可将来自所插模板的在线信息与所组态模板的数据进行比较
诊断缓冲区	诊断缓冲区中事件一览表以及所选中事件的详细信息	使用诊断缓冲区还可在晚些时候对系统错误进行分析，查找停机原因并对出现的每个诊断事件分类
诊断中断	所选模板的诊断数据	用于评估模板故障的原因
DP 从站诊断	所选 DP 从站（参照 EN50170）的诊断数据	评估 DP 从站中的故障原因
存储器	存贮能力所选的 CPU 或 M7 功能模板的工作存储器和装载存储器当前的使用情况	在传送新的块或扩展了的块传到 CPU 之前，可检查 CPU/功能模板的装载存储器中是否有足够的空间，或用于压缩存储器内容
扫描循环时间	所选 CPU 或 M7 功能模板最长、最短和最近的循环扫描时间	用于检查所组态的最小循环时间，最大循环时间和当前循环时间
时间系统	当前时间，操作小时数以及关正同步时钟的信息（同步周期）	可显示并设置模板的时间和日期，检查时间同步
性能数据	地址区域和所选模板（CPU/FM）可以使用的块	在生成用户程序之前或之中检查 CPU 是否满足执行用户程序的要求。例如，装载存储区的大小或过程映象的大小
块(可在“ Performance Data ”选项卡中打开)	显示所选模板提供范围内所有可用的块类型，包括 OB、SFB、SFC 列表，你可将这些块用于该模板	检查你的用户程序中可包含或调用哪些标准块在所选 CPU 上运行
通讯	传送速率，通讯连接概述，通讯负载以及所选模板在通讯总线上最大的信息的容量	用来决定有多少以及哪种 CPU 或 M7FM 的连接是可能的；有多少可处于使用中
堆栈	堆栈选项卡：只能在 STOP 模式或 HOLD 模式下调用 显示所选模板的 B（块）堆栈。然后你还可以显示 I（中断）堆栈，L（局域）堆栈以及嵌套深度堆栈。可以跳转到中断块的故障点。	可判明引起停机的原因并对块进行修改

附加的信息显示

对每个选项卡可显示以下信息：

- 到所选模板的在线路经。
- 相应CPU的操作模式（例如：运行、停机）
- 所选模板的状态（例如：出错、ok）
- 所选模板的操作模式（例如：运行、停机），如果它有自己的操作模式的话（如CP342-5）。

在从“ Accessible Nodes ”窗口打开的非 CPU 模板的模板信息中，CPU 本身的操作模式和所选模板的状态不能显示。

同时显示许多模板

你可以同时显示许多模板的模板信息。要做到这一点，你必须到各个模板的上下文中，选择另一个模板，然后为它调用模板信息。“ Module Information（模板信息）”对话框就会显示出来。为每一个模板只能打开一个对话框。

刷新模板的信息显示

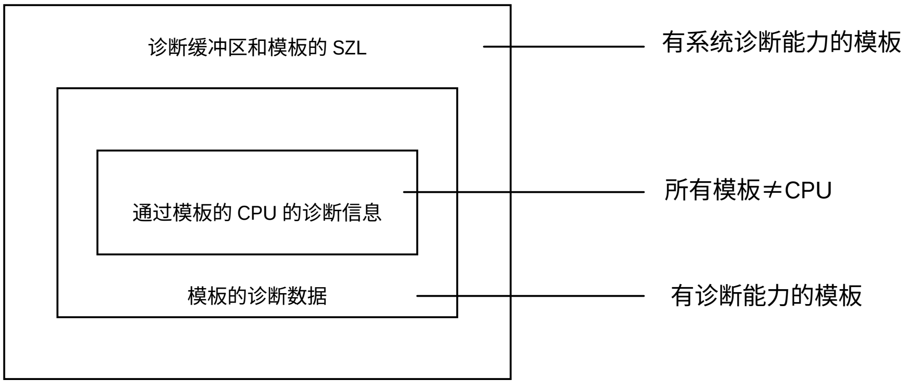
每次在“ Module Information ”对话框中转换选项卡时，数据都要再从模板上读过来。但是，当一页显示之后，其内容不再刷新。如果你点击“ Updete（刷新）”按钮，可以在不改变选项卡的情况下从模板读数据。

21.5.3 依模板类型而不同的模板信息的范围

可以评估和显示的信息范围依赖以下几点：

- 所选模板，以及
- 从哪个视窗调用模板信息
  当从组态表的在线视窗或项目窗口调用时可得到全范围的信息。
  从“ Accessible Nodes（可访问站）”窗口调用只能得到有限范围内的信息。

根据信息的范围，模板可分为几类：“ 有系统诊断能力 ”、“ 有诊断能力 ” 或 “ 无诊断能力 ”。下图所示为这些类别：



- 有系统诊断能力的模板是，如模板FM351和FM354
- 有诊断能力的模板是大多数的模拟信号模板
- 没有诊断能力的模板是大多数的数字信号模板

选项卡显示

下表显示了 “ 模板信息 ” 对话框中每个类型的模板都显示哪些特性选项卡。

选项卡	CPU 或 M7 FM	有系统诊断能力的模板	有诊断能力的模板	无诊断能力的模板	DP 从站
常规	有	有	有	有	有
诊断缓冲区	有	有	---	---	---
诊断中断	---	有	有	---	有
存储器	有	---	---	---	---
扫描循环时间	有	---	---	---	---
系统时间	有	---	---	---	---
性能数据	有	---	---	---	---
堆栈	有	---	---	---	---
通讯	有	---	---	---	---
DP 从站诊断	---	---	---	---	有
H 状态 [1)]	有	---	---	---	---
1） 只有 H 系统的 CPU					

除了选项卡特性页的信息之外，有操作模式的模板还会显示操作模式。当你从在线的组态表中打开对话框时，来自 CPU 的视角的模板状态会被提示（例如，OK、故障、模板不存在）。

21.6 在停机模式下诊断

21.6.1 判定停机原因的基本程序

要判断 CPU 为什么进入“停机”模式，可按如下进行：

1. 选择已进入停机的CPU。
2. 选择菜单命令PLC＞Module Information。
3. 选择“Diagnostic Buffer（诊断缓冲区）”选项卡。
4. 你可以从诊断缓冲区中最后一项判定停机的原因。

如果有编程错误出现：

1. 输入项“STOP because programming error OB not loaded”意味着，CPU 已查到一个编程错误而且试图启动这个（不存在的）OB 块去处理这个编程错误。前一条指出了实际的编程错误。
2. 选择与编程错误相关的信息。
3. 点击“Open Block”按钮。
4. 选择“Stacks（堆栈）”选项卡。

21.6.2 停机模式下堆栈的内容

通过评估诊断缓冲区和堆栈中的内容，你可以判定用户程序执行过程中引起故障的原因。

例如，如果由于编程错误或停机指令使 CPU 进入停机状态。你可以用“I Stack（中断堆栈）”“L Stacks（局域堆栈）”和“Nesting Stack（嵌套堆栈）”按钮显示这些堆栈中的内容。堆栈内容为你提供哪个块中的哪条指令引起 CPU 进入停机的信息。

B堆栈内容

B 堆栈或称作块堆栈，列出了所有停机前已经被调用但还未完全处理完的块。

I栈内容

当你点击“ I stack（中断堆栈） ”的按钮时，中断点的数据则被显示。这个 I 堆栈，或称中断堆栈包含着中断时有效的数据或状态，例如：

- 累加器内容和寄存器内容
- 打开的数据块和他们的大小
- 状态字的内容
- 优先级（嵌套层次）
- 中断的块
- 中断后程序将继续处理的块

L堆栈内容

对于每个列在 B 堆栈中的块，都可以通过选择该块并点击“ L Stack（局域堆栈） ”按钮显示相应的局域数据。

这个 L 堆栈，或称作局域数据堆栈，包含中断时用户程序正在工作的块的局域数据。

解释和评估所显示的局域数据需要更深入地系统知识。显示的第一部分的数据相应于块中的临时变量。

嵌套堆栈内容

当你点击“ Nesting Stack（嵌套堆栈） ”按钮时，显示嵌套堆栈在断点处的内容。

嵌套堆栈是逻辑操作 A(，AN(，O(，ON(，X(和 XN(使用的存储区域。

只有当中断时有括号操作仍在打开，该按钮才激活。

21.7 检查扫描循环时间以免除时间错误

21.7.1 检查扫描循环时间以免除时间错误

在模板信息的“Scan Cycle Time（循环扫描时间）”选项卡中可以给出有关用户程序扫描循环时间的信息。

如果最长的循环时间接近组态的最大扫描循环时间，就会存在由于循环时间的波动引起时间错误的危险。如果你延长用户程序的最大循环时间（监控时间）则可以避免这种危险。

如果循环长度短于组态的最小循环时间，则由 CPU/FM 自动延长循环至组态的最小循环时间。对于 CPU，在这个延长时间内背景 OB（OB90）会被处理（如果它已下装）。

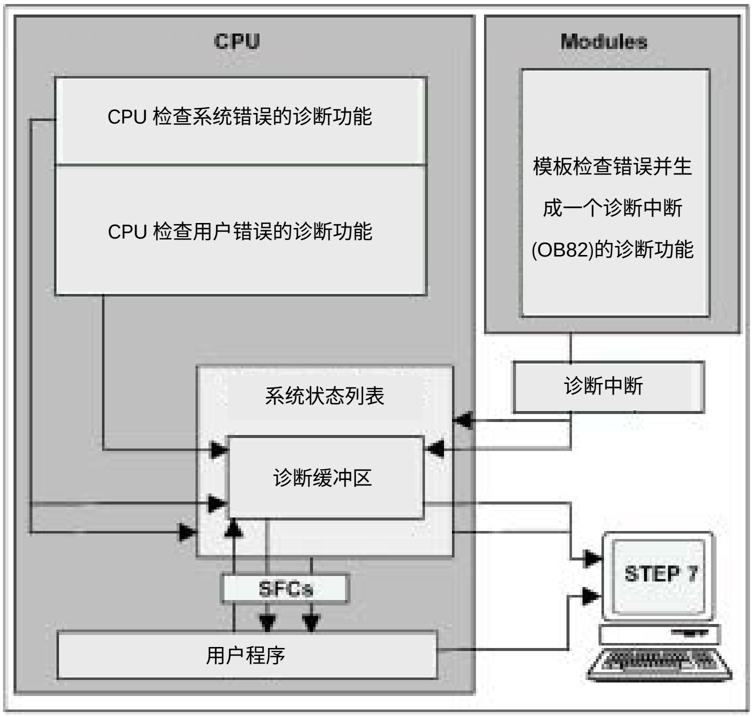
设置扫描循环时间

当你组态硬件时，可以设置最大和最小循环时间。要做这一步，双击组态表离线视窗中的CPU/FM定义它的特性。你可以在“Cycle/Clock Memory（循环/时钟存储器）”选项卡中输入适当的值。

21.8 诊断信息的流动

21.8.1 诊断信息的流动

下图所示为在 SIMATIC S7 中诊断信息的流动



显示诊断信息

你可以在用户程序中使用 SFC 51 RDSYSST 读出诊断内容，或者用 STEP 7 以无格式语言显示诊断信息。

它们可以提供以下信息：

- 错误出现的位置和时间
- 该输入项所属的诊断事件的类型（用户定义的诊断事件、同步/异步错误、操作模式改变）。

生成控制组报文

CPU 在诊断缓存区中送入标准诊断事件和扩展诊断事件。如果满足下列条件，它还为标准诊断事件生成一个过程控制组报文：

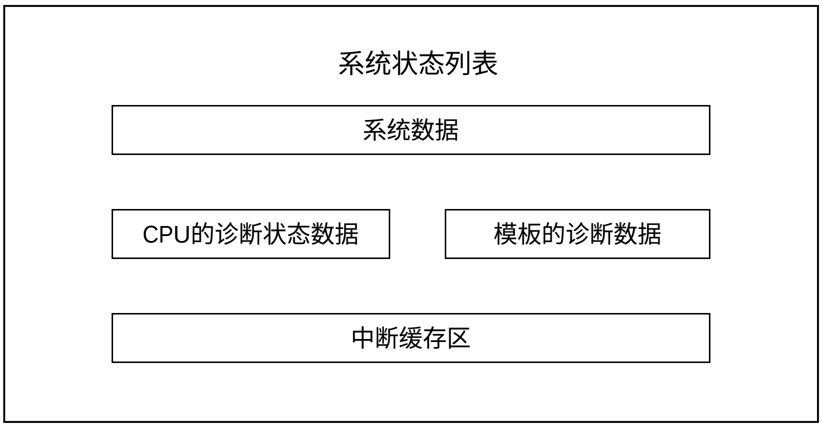
- 你已指定过程控制报文将在STEP 7中生成。
- 至少有一个过程控制报文的显示单元已在CPU上登录。
- 一个过程控制组报文生成的条件是当前没有相应级别的过程控制组报文（有七个级别）。
- 每个级别可生成一个过程控制组报文。

21.8.2 系统状态列表 SSL

系统状态列表（SSL）描述可编程控制器的当前状态。它提供了组态、当前参数赋值、当前 CPU 的状态和次序以及所属模板的概观。

系统状态列表中的数据只能读不能修改。它是一个实际的列表，只当有请求时才会生成。

使用系统状态列表能够显示的信息可分为四个区域。



读出系统状态列表

如下所示有两种读出系统状态列表的方法：

- 间接地，通过编程设备中STEP 7的菜单命令（例如，存储器组态、静态CPU数据、状态显示）。
- 直接地，在用户程序中通过系统功能SFC51 RDSYSST，输入所需的部分系统状态列表的号码（见关于该块的帮助）。

系统状态列表的系统数据

系统数据是 CPU 内固有的数据或赋值的性能数据。下表所示是可以显示的信息的题目（部分系统状态列表）：

标　题	信　息
模板标识	定货号、类型标识以及模板的版本
CPU 特性	系统时间、系统行为（如：多 CPU）以及对 CPU 的语言描述
存储区	模板的存储器组态（工作存储器的大小）
系统存储区	模板的系统存储器（例如：存储位、定时器、计数器的数量、存储器的类型）
块类型	模板中存在哪些块（OB、DB、SDB、FC、FB），某一类型块的最大数量，以及某一类块的大小
中断及错误的赋值	中断/错误与 OB 之间的赋值关系
中断状态	产生的中断处理/中断当前的状态
优先级的状态	由参数设定的哪个 OB 块被执行，哪个优先级被禁止
操作模式和模式转换	最后的操作模式和当前操作模式都可以

CPU中的诊断状态数据

诊断状态数据描述了由系统诊断监视的组件的当前状态。下表所示是可以显示的信息的题目（部分系统状态列表）：

标　题	信　息
通讯状态数	当前在系统中设置的所有通讯功能
诊断模板	在 CPU 中登录了的有诊断能力的模板
OB 的启动信息列表	有关 CPU 中 OB 的启动信息
启动事件列表	OB 的启动事件及优先级
模板状态信息	关于插入的已赋值的所有模板的状态信息，故障或产生硬件中断

模板上的诊断数据

除 CPU 之外，还有其它的模板具有诊断能力（SM、CP、FM），它们的数据被存入系统状态列表。下表所示为能够显示的信息的题目（部分系统状态列表）：

标　题	信　息
模板诊断信息	模板起始地址、内部/外部故障、通道故障、参数错误（4 字节）
模板诊断数据	某个特定模板的所有诊断数据

21.8.3 传送你自己的诊断报文

你可以用系统功能 SFC 52 WRUSMSG 扩展 SIMATIC S7 的标准系统诊断：

- 在诊断缓存区中输入你自己的诊断信息（例如，关于用户程序执行的信息）。
- 传送用户定义的诊断报文到已登录的站（监视设备如PG、OP或TD）。

用户定义的诊断事件

诊断事件可以被分为事件级别 1 至 F。用户定义的诊断事件属于事件级别 8 至 B。可按如下所示分为两组：

- 事件级别8和9包括那些有固定编号和预定文本的报文，你可以根据编号调用。
- 事件级别A和B包括那些可以由你自己选择分配一个号码（A000至A0FF，B000至B0FF）和文本的报文。

传送诊断报文到站

除了可以在诊断缓存区存入用户定义的输入项，你还可以用 SFC 52 WRUSMSG 将你自己的用户定义的诊断报文传送到已登录的显示设备上。当用 SEND=1 调用 SFC 52 时，诊断报文被写入传送缓冲区并自动发送到在 CPU 上登录过的站。

如果无法传送报文（比如，因为没有登录的显示设备或由于传送缓冲区满了），用户定义的诊断事件仍然存在诊断缓存区中。

生成一个带应答的报文

如果你应答了一个用户定义的诊断事件并且想要记录这个应答，可按如下进行：

- 当事件到来时事件状态将1写入一个BOOL类型的变量，当事件离开时事件状态将0写到这个变量。
- 然后你可以用SFB33 ALARM监视这个变量。

21.8.4 诊断功能

系统诊断检测、评估和报告可编程控制器内出现的错误。为这一目的，每个 CPU 和每个有系统诊断能力的模板（如 FM354）都有一个诊断缓冲区，在这个缓冲区内诊断事件的详细信息按它们出现的顺序存入。

诊断事件

下列事项将作为诊断事件显示，例如：

- 模板上的内部和外部错误
- CPU中的系统错误
- 操作模式改变（如，从RUN到STOP）
- 用户程序中的错误
- 插入/移走模板
- 用系统功能SFC52输入的用户报文

在存储器全清后诊断缓存区中的内容仍然保留。使用诊断缓冲区还可在晚些时候对系统错误进行分析，查找停机原因并对出现的每个诊断事件分类

获取诊断数据

你不必为获取系统诊断的诊断数据而编写程序。这是一个可自动运行的标准特性。SIMATIC S7 提供各种诊断功能。一些诊断功能是集成在 CPU 上的，另一些由模板提供（SM、CP 和 FM）。

显示故障

内部和外部模板故障会显示在模板的前面板上。有关 LED 的显示以及对它们的评估的描述在 S7 硬件手册中。在 S7-300 中，内部和外部故障作为一个组错误显示。

CPU 能够识别系统错误和用户程序错误并将诊断信息存入到系统状态列表和诊断缓存区中。这些诊断信息可以由编程设备读出。

具有诊断能力的信号模板和功能模板可检测内部和外部模板错误并产生一个诊断中断，你可以用一个中断 OB 来响应这个中断。

21.9 程序测试

21.9.1 程序测试

当 CPU 检测到程序处理过程中的错误（同步错误）和可编程控制器中的错误（异步错误）时，CPU 会调用适当的组织块（OB）：

错　误	错误 OB
I/O 冗余错误	OB 70
CPU 冗余错误	OB 72
通讯冗余错误	OB 73
时间错误	OB 80
电源错误	OB 81
诊断中断	OB 82
插入/移出模板中断	OB 83
CPU 硬件故障	OB 84
优先级错误	OB 85
机架故障或分布式 I/O 的站故障	OB 86
通讯错误	OB 87
编程错误	OB 121
I/O 访问错误	OB 122

如果相应的 OB 不存在，CPU 进入 STOP 模式。否则，可在 OB 中存储指令以决定对这种错误作何反应。这意味着错误的影响可以减小或根除。

基本程序

生成并打开 OB

1. 显示 CPU 的模板信息。
2. 选择“ Performance Data（性能数据） ”选项卡。
3. 在所显示的列表的基础上，决定你要编程的 OB 是否能在该 CPU 上执行。
4. 在你的程序的“ Block（软件块）文件夹中插入 OB 并打开该 OB 块。
5. 为故障处理输入程序。
6. 将 OB 下载到可编程控制器。

故障处理的编程措施

1. 评估 OB 块的局域数据以判断故障的确切原因。局域数据中的变量 OB8xFLTID 和 OB12xSWFLT 中包含错误代码。有关它们的含义的描述在《系统和标准功能参考手册》里。
2. 转向响应故障的程序段

在标头为“ 用 SFC51 （RDSYSST）作模板诊断的示例 ”的关于系统和标准功能的在线帮助中，你能够找到处理诊断中断的示例。

在相应的块帮助中可以找有关 OB、SFB、SFC 的更详细的信息。

21.9.2 评估输出参数 RET_VAL

系统功能用输出参数RET_VAL（返回值）指示CPU是否正确地执行了SFC功能。

返回值中的错误信息

返回值是整数类型（INT）。整数的符号位指示该整数是正还是负。返回值与数值“ 0 ”之间的关系用以指示在功能执行过程中是否有错误出现（见下表）：

- 如果功能执行时出现错误，返回值小于“ 0 ”。整数的符号位是“ 1 ”。
- 如果功能执行时没有错误，返回值大于等于“ 0 ”。整数的符号位是“ 0 ”。

由 CPU 处理 SFC	返回值	整数的符号
错误出现	小于“ 0 ”	负（符号位是“ 1 ”）
没有错误	大于等于“ 0 ”	正（符号位是“ 0 ”）

对故障信息的反应

如果在 SFC 执行时出现错误，SFC 在返回值（RET_VAL）中提供一个错误代码。

作以下区分：

- 所有SFC都可以输出的通用错误代码以及
- 依据SFC的特定功能而输出的特定错误代码

传送功能值

有些 SFC 还用输出参数 RET_VAL 来传送功能值，例如 SFC64TIMETCK 用 RET_VAL 传送它已读出的系统时间。

在 SFB/SFC 帮助中可以找到一些有关输出程序 SFB/SFC 更详细的信息。

21.9.3 故障 OB 作为对已检测错误的响应

可检测的错误

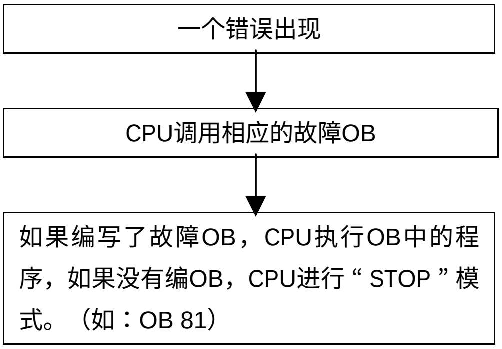
系统程序可以检测以下错误：

- 不正确的CPU功能
- 系统程序执行中的错误
- 用户程序中的错误
- I/O中的错误

根据错误类型的不同，CPU 被设定为 STOP 模式或调用一个故障 OB。

编程响应

你可以设计程序响应不同类型的错误，并决定 CPU 的反应方式。为某个特定的错误而编制的程序可以保存在一个故障 OB 中。如果故障 OB 块被调用，程序就会被执行。



故障OB

按如下所示区别同步错误和异步错误：

- 同步错误可以分配给一个MC7指令（例如，对一个已移走的信号模板作装载操作）。
- 异步错误可分配给一个优先级或给整个可编程控制器（例如循环超时）。

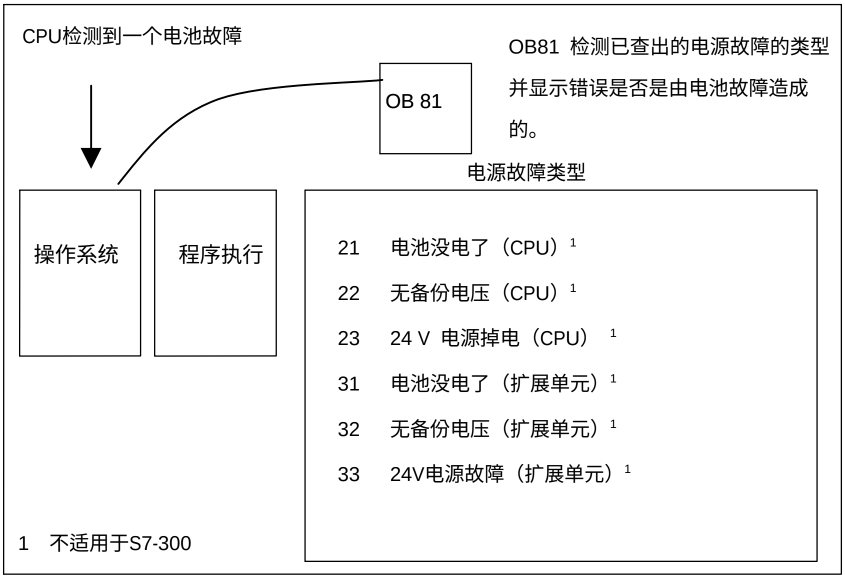
下表所示为可能出现的错误类型。参考你的《S7-300 可编程控制器，硬件及安装手册》或《S7-400，M7-400 可编程控制器，硬件及安装手册》，可找到有关你的 CPU 是否支持这些特定的 OB 的信息。

错误级别	错误类型	OB	优先级
冗余	I/O 冗余错误(只在 H CPU 中)	OB 70	25
	CPU 冗余错误(只在 H CPU 中)	OB 72	28
	通讯冗余错误	OB 73	25
异步的故障	时间错误	OB 80	26
	电源错误	OB 81	（或者 28 如果故障 OB 在启动程序中调用了）
	诊断中断	OB 82	启动程序
	插入/移出模板中断	OB 83	
	CPU 硬件故障	OB 84	
	程序顺序错误	OB 85	
	机架坏了	OB 86	
	通讯错误	OB 87	
同步	编程错误	OB 121	引起错误的 OB 的优先级
	I/O 访问错误	OB 122	

故障OB81使用举例

使用错误 OB 的局域数据（启动信息），可以判断已出现的错误的类型。

例如，如果 CPU 检测到一个电池故障，操作系统则调用 OB81（见图）。



你可以编写一个程序来判定由于 OB81 的调用而触发的事件代码。也可以编写程序作出响应，如，激活与操作站上的指示灯相连的输出。

故障OB81的局域数据

下图所示为在这种情况下必须在 OB81 的变量声明表中进行声明的临时变量。

符号“ Battery error（BOOL）”必须标识为输出（例如，Q4.0），这样程序的其它部分就能够访问这些数据了。

声明	名称	类型	描述
TEMP	OB81EVCLASS	BYTE	错误级别/错误标识 39xx
TEMP	OB81FLTID	BYTE	错误代码： b#16#21 = CPU 中至少有一个备份电池没电了 1) b#16#22 = 在 CPU 中无备份电压 b#16#23 = CPU 中 24-V 电源掉电 1) the CPU 1 b#16#31 = 至少有一个扩展机架上的一个备份电池没电了 1) b#16#32 = 在一个扩展机架上没有备份电压 1) b#16#33 = 一个扩展机架 24V 电源掉电 1)
TEMP	OB81PRIORITY	BYTE	优先级=26/28
TEMP	OB81OBNUMBR	BYTE	81 = OB81
TEMP	OB81RESERVED1	BYTE	保留
TEMP	OB81RESERVED2	BYTE	保留
TEMP	OB81MDLADDR	INT	保留
TEMP	OB81RESERVED3	BYTE	只与错误代码 B#16#31，B#16#32，B#16#33 相关
TEMP	OB81RESERVED4	BYTE	
TEMP	OB81RESERVED5	BYTE	
TEMP	OB81RESERVED6	BYTE	
TEMP	OB81DATETIME	DATEAND TIME	启动 OB 的日期和时间
1）=不适用于 S7-300			

故障OB81程序举例

在 STL 程序举例中说明了如何在 OB81 中读错误代码。

程序结构如下：

- OB81中的错误代码（OB81 FLTID）被读出并与事件“ 电池没电 ”（B#16#3921）的数值作比较。
- 如果错误代码符合“ （battery exhausted）电池没电 ” 的代码，程序则跳到标号为Berr的指令并激活输出*batteryerror*。
- 如果错误代码与“ （battery exhausted）电池没电 ” 的代码不符，程序则将错误代码与“ 电池故障 ” 的代码作比较。
- 如果错误代码符合“ 电池故障 ” 代码，程序跳转到标号“ Berr ”，并激活输出“ *batteryerror* ” 。否则该块结束。

AWL		说明
L	B#16#21	// 比较事件代码“ 电池没电 ”
		// （B#16#21）
L	#OB81_FLT_ID	// 与OB81的错误代码。
==I		// 如果相同（电池没电），
		// 跳转到Berr。
JC Berr		
L	B#16#21	// 比较事件代码“ 电池故障 ”
		//（b#16#22）
==I		// 与OB81的错误代码。
JC BF		// 如果不相同，跳转到 Berr。
BEU		// 无电池故障信息
Berr：L	B#16#39	// 与下一事件ID进行比较
L	#OB81_EV_CLASS	// OB81的错误代码。
==I		// 如果电池故障或电池没电找到，
S	batteryerror	// 设置输出“ battery error ” 。
		//（符号表变量）
L	B#16#38	// 比较ID，以确定OB81
==I		// 错误代码事件。
R	batteryerror	// 复位输出“ battery error ”，当错误固定时。

在相应的块帮助中可以找有关 OB、SFB、SFC 的更详细的信息以及事件 ID 解释。

21.9.4 为故障诊断插入替代值

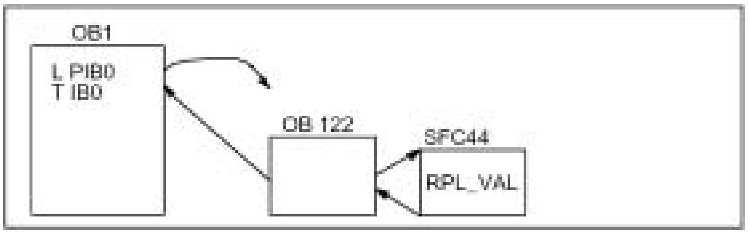
对于某种类型的故障（如，受断线影响的输入信号），你可以为由于故障而无法使用的数值提供一个替代位。可用以下两种方法来提供替代值：

- 用STEP 7为可组态的输出模板分配替代值。无法得到赋值参数的输出模板用缺省替代值。
- 用SFC44 RPLVAL，可以在故障OB中编程替代值（只适用于输入模板）。

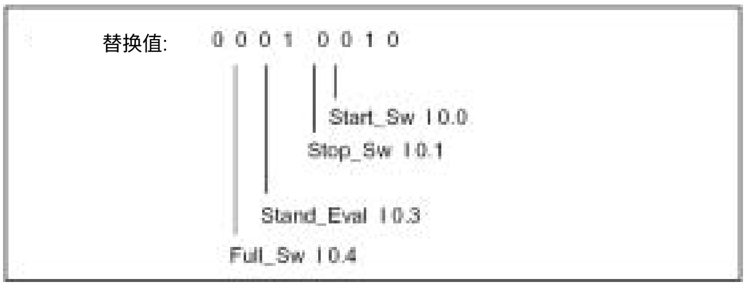
对于所有引起同步错误的装载指令，你可以在故障 OB 中为累加器内容指定一个替代值。

替代数值程序举例

在以下示例程序中，在 SFC44RPLVAL 中有一个可用的替代值。下图说明了 CPU 是如何在检测到一个输入模板没有反应时调用 OB122 的。



在这个示例中，下图所示的替代值在程序中被输入，这样程序就可以用可行的数值继续操作。



如果一个输入模板有故障，执行指令 L PIB0 就会产生一个同步错误并启动OB122。作为标准，这个装载指令读得数值 0。然而，用 SFC44，你可以为过程定义任何合适的值。SFC 用指定的替代值替换累加器中的内容。

以下示例程序可写在 OB122 中。下表所示为在这种情况下，在 OB122 的变量声明表中必须声明的临时变量。

声明	名称	类型	描述
TEMP	OB122EVCLASS	BYTE	错误级别/错误 ID29xx
TEMP	OB122SWFLT	BYTE	错误代码： 16#42， 16#43， 16#44 1)，16#45 1)
TEMP	OB122PRIORITY	BYTE	优先级=错误出现的 OB 的优先级
TEMP	OB122OBNUMBR	BYTE	122 = OB122
TEMP	OB122BLKTYPE	BYTE	错误出现的块的类型
TEMP	OB122MEMAREA	BYTE	存储区域和访问类型
TEMP	OB122MEMADDR	WORD	出错的存储器地址
TEMP	OB122BLKNUM	WORD	出错的块的号码
TEMP	OB122PRGADDR	WORD	出错指令的相对地址
TEMP	OB122DATETIME	DATEANDTIME	启动 OB 的日期和时间
TEMP	错误	INT	存储 SFC44 的错误代码
1）不适用于 S7-300			

STL	说明
L B#16#2942	为应答读I/O时出现的时间错误，将OB122的事件码与事件码
L	（B#16#2942）比较，
#OB122SWFLT	如果一样跳到“ Aerr ”。
==I	为寻址错误（向一个不存在的模板作写操作）将OB122的事件代
JC Aerr	码与事件代码（B#16#2943）作比较。如果一样跳到“ STOP ”。
L B#16#2943	标号“ Aerr ”：将DW#16#2912（二进制10010）传送到SFC44
<> I	（REPL_VAL）。SFC 44将该值装载到累加器1（替代由OB122
JC Stop	调用而触发的数值）。SFC的错误代码存储在#Error。
Aerr： CALL ”REPL_VAL"	
VAL ：= DW#16#2912	将#Error与0比较（如果相同则OB122执行时没出错）。
RETVAL ：= #Error	如果没有错结束块。
L #Erro	
L 0	“ Stop ”标号：调用SFC46“ STP ”，
==I	将CPU转为STOP模式。
BEC	
Stop： CALL “STP"	

21.9.5 I/O 冗余错误（OB70）

说明

对于 H CPU 的操作系统，如果 PROFIBUS DP 上出现冗余丢失（如，DP 主站总线故障或 DP 从站接口模板故障），或者，如果带有转换 I/O 的 DP 从站变为激活的 DP 主站时，系统调用 OB70。

编程OB70

你必须用 STEP 7 在用户程序中将 OB70 作为一个对象生成。在生成的块中编写要在 OB70 中执行的程序，并作为你的用户程序的一部分下载到 CPU。

例如，你可以为如下目的使用 OB70：

- 要评估OB70中的运动信息并判定哪个条件触发了I/O冗余丢失。
- 用SFC51RDSYSST判定你的系统的状态。（SZLID=B#16#71）。

如果出现 I/O 冗余错误且 OB70 没有编程，CPU 不停机。

如果 OB70 下载了并且 H 系统不在冗余模式，OB70 在两个 CPU 中都被处理。H 系统保持在冗余模式。

在相应的块帮助中可以找有关 OB、SFB、SFC 的更详细的信息。

21.9.6 CPU 冗余错误（OB72）

说明

如果出现以下事件之一，H CPU 的操作系统调用 OB72：

- CPU冗余丢失
- 比较错误（如RAM、PIQ）
- 待机主站切断
- 同步错误
- SYNC子模块出错
- 刷新过程被放弃
- OB72被所有的CPU执行， CPU在一个伴随的起动事件后，处于RUN或STARTUP模式。

编程OB72

必须使用 STEP 7 在你的 S7 用户程序中将 OB72 作为一个对象生成。在生成的块中编写将由 OB72 执行的程序并将它作为你的用户程序的一部分下载到 CPU。

例如，你可以为以下目的使用 OB72：

- 要评估OB72中的起动信息和判定哪个事件触发了CPU冗余的丢失。
- 用SFC51RDSYSST判定你的系统的状态。（SZLID=B#16#71）。
- 要为系统对CPU的冗余丢失作出特定的反应。

如果出现 CPU 冗余错误，并且 OB72 没有编程，CPU 不会转为 STOP 模式。

在相应的块帮助中，可以找有关 OB、SFB、SFC 的更详细的信息。

21.9.7 通讯冗余错误（OB73）

说明

H CPU 操作系统可以在冗余 S7 连接中有冗余损坏时调用 OB73。（冗余 S7 连接排它性地存在于 S7 连接中；见“S7-400 H 可编程器控制器冗余系统”）。如果有其它冗余 S7 连接的冗余丢失，OB73 不能再启动。

只有所有冗余 S7 连接的冗余都恢复，OB73 才能启动。

如果出现相应的开始事件并且 OB73 没有编程，CPU 不会转为 STOP 模式。

编程OB73

必须用 STEP 7 在你的 S7 程序中将 OB73 作出一个对象生成。在生成块中编写要在 OB73 中执行的程序，并将它作为你的用户程序的一部分下载到 CPU。

例如，你可以为以下目的使用 OB73：

- 要评估OB73中的起动信息并判定哪个条件触发了I/O冗余丢失。
- 用SFC51RDSYSST判定你的系统的状态（SZL_ID=B#16#71）。

如果出现通讯冗余错误并且 OB73 没有编程，CPU 不会转为 STOP 模式。

如果 OB73 下载了并且 H 系统处于冗余模式，OB73 在两个 CPU 中都被执行。H 系统保持在冗余模式。

在相应的块帮助中可以找有关 OB、SFB、SFC 的更详细的信息。

21.9.8 时间错误（OB80）

说明

当有时间错误出现时 CPU 的操作系统调用 OB80。时间错误包括以下，如：

- 超过最大循环时间
- 通过向前修改时间而跳过日时钟中断
- 处理优先级时延迟太多

编程OB80

必须用 STEP 7 在你的 S7 程序中将 OB80 作为一个对数生成。在生成的块中编写要在 OB80 中执行的程序并将它作为你的用户程序的一部分下载到 CPU。

例如，可为以下目的使用 OB80：

- 要评估OB80中的起动信息和判定哪个日时钟中断被跳过。
- 通过使用SFC29 CANTINT，可以取消已被跳过的日时钟中断而不再执行它，只有与新的时间相关的日时钟中断全被执行。

如果你没有在 OB80 中取消跳过的日时钟中断，则第一个跳过的日时钟中断被执行，所有其它的被忽略。

如果你没有编程 OB80，当 CPU 检测量到一个时间错误时转为 STOP 模式。

在相应的块帮助中可以找有关 OB、SFB、SFC 的更详细的信息。

21.9.9 电源故障（OB81）

说明

如果在 CPU 中或在扩展单元中有以下故障出现，CPU 的操作系统调用 OB81

- 24V电压提供
- 电池
- 完全备份

当问题消除时 OB 也会被调用（OB 在事件到来和离开时被调用）。

编程OB81

必须用 STEP 7 在你的 S7 程序中将 OB81 作为一个对象生成。在生成的块中编写要在 OB81 中执行的程序并将它作为你的用户程序的一部分下载到 CPU。

例如，你可以为如下目的编写 OB81：

- 要评估OB81的起动信息以及判定出现哪个电源故障。
- 要查找有电源故障的机架的号码。
- 要激活操作站上的灯指示维护人员应该换电池了。

与其它的异步故障 OB 相反，如果没有编写 OB81，CPU 检测到电源故障时不会转为 STOP 模式。但是这个错误会存入诊断缓冲区并且前面板上相应的 LED 灯亮，指示错误。

在相应的块帮助中可以找有关 OB、SFB、SFC 的更详细的信息。

21.9.10 诊断中断（OB82）

说明

对于一个有诊断能力的模板，如果你使能了它的诊断中断，当它检测到错误，以及错误消除时 CPU 的操作系统会调用 OB82。（该 OB 在事件到来和离去时都会被调用）。

编程OB82

必须用 STEP 7 在你的 S7 程序中将 OB82 作为一个对象生成。在生成的块中编写要在 OB82 中执行的程序，并将它作为你的用户程序的一部分下载到 CPU 中。

例如，你可为以下目的使用 OB82：

- 要评估OB82的起动信息。
- 要获得与已出现的错误有关的更确切的诊断信息。

当一个诊断中断被触发时，有问题的模板自动地在诊断中断 OB 的起动信息和诊断缓冲区中存入 4 个字节的诊断数据及其起始地址。这就为你提供了错误何时出现以及出现在哪个模板上的信息。

在 OB82 中编写合适的程序，你可以进一步地评估模板的诊断数据（哪个通道出错，出现的是哪种错误）。使用 SFC51 RDSYSST 可以读出模板的诊断数据，用 SFC52WRUSRMSG 可以将这些信息存入诊断缓冲区。你也可以发送一个用户定义的诊断报文到监控设备。

如果没有编写 OB82，当诊断中断被触发时 CPU 转为 STOP 模式。

在相应的块帮助中可以找有关 OB、SFB、SFC 的更详细的信息。

21.9.11 插入/移开模块中断（OB83）

说明

S7-400CPU 以大约 1 秒的间隔监视着中央机架和扩展机架上现有的模板。

在电池上电后，CPU 检测由 STEP 7 生成的组态表中所列的所有模板是否都实际地插入了。如果所有模板都有，这个实际组态被存储并用作对模板进行循环监控的参考值。在每一扫描循环，新检测到的实际组态与原来的实际组态作比较。如果发现两个组态有差异，则发出插入/移开模板中断信号，并且有信息存入诊断缓冲区和系统状态列表。在 RUN 模式下，启动插入/移开模板中断 OB。

注意

电源模板、CPU 和 IM 不能在 RUN 模式下移开。

在移走和插入模板两个操作之间应至少相隔两秒，以便让 CPU 检测到移走的或插入的模板。

对新插入的模板进行参数赋值

如果一个模板在 RUN 模式下插入，CPU 会检测新模板的类型与原来模板是否匹配。如果匹配，对该模板赋予参数。缺省参数或用 STEP 7 分配的参数被传送到模板中。

编程OB83

必须使用 STEP 7 在你的 S7 程序中将 OB83 作为一个对数生成。在生成的块中编写要在 OB83 中执行的程序，并将它作为你的用户程序的一部分下载到 CPU 中。

例如，你可以为以下目的使用 OB83：

- 要评估OB83的起动信息。
- 用系统功能SFC55至59对新插入的模板赋值参数。

如果没有编写 OB83，为一个插入/移开模板中断出现时，CPU 从 RUN 转为 STOP。

在相应的块帮助中可以找有关 OB、SFB、SFC 的更详细的信息。

21.9.12 CPU 硬件故障（OB84）

说明

当 CPU 检测到连至 MPI 网络的接口故障、连至通讯总线的接口故障或连至分布式 I/O 网卡的接口故障时操作系统调用 OB84；例如，在总线上检测到一个不正确的信号层。故障消除时也会调用该 OB 块（事件到来和离开时都调用该 OB）。

编程OB84

必须使用 STEP 7 在你的 S7 程序中将 OB84 作为一个对象生成。在生成的块中编写要在 OB84 中执行的程序并将它作为你的用户程序的一部分下载到 CPU。

你可以为以下目的使用 OB84：

- 要评估OB84的起动信息。
- 用系统功能SFC52 WRUSMSG发送报文至诊断缓存区。

如果 OB84 没有编程，当检测到 CPU 硬件错误时 CPU 转为 STOP 模式。

在相应的块帮助中可以找有关 OB、SFB、SFC 的更详细的信息。

21.9.13 编程顺序错误（OB85）

说明

CPU 的操作系统调用 OB85：

- 当一个中断OB的起动事件存在，但该OB块由于没有下载到CPU而不能被执行时。
- 当访问一个系统功能块的背景数据块时出错。
- 当刷新过程映象表时出错（模板不存在或出故障）。

编程OB85

必须用 STEP 7 在你的 S7 程序中将 OB85 作为一个对象生成。在生成的块中编写要在 OB85 中执行的程序并将它作为你的用户程序的一部分下载到 CPU。

例如，你可以为以下目的使用 OB85：

- 要评估OB85的起动信息和判定哪个模板损坏或没插入（指定模板的起始地址）。
- 用SFC49 LGCGADR查找相关模板所在的槽。

如果没有编写 OB85，当优先级错误被检测到时 CPU 转为 STOP 模式。

在相应的块帮助中可以找有关 OB、SFB、SFC 的更详细的信息。

21.9.14 机架故障（OB86）

说明

当检测到机架故障时，CPU 的操作系统调用 OB86，如：

- 机架故障（找不到IM或IM损坏，或者连接电缆断线）
- 机架上的分布电源故障
- 在SINEC L2-DP总线系统的主系统中有一个DP从站出故障

故障消除时也会调用该 OB 块（事件到来和离开时都调用该 OB）。

编程OB86

必须用 STEP 7 在你的 S7 程序中将 OB86 作为对象生成。在生成的块中编写要在 OB86 中执行的程序，并将它作为你的用户程序的一部分下载到 CPU 中。

例如，你可以为以下目的使用 OB86：

- 要评估OB86的起动信息和判定哪个机架损坏或找不到。
- 用系统功能SFC 52 WRUSNSG将报文存入诊断缓冲区，并发送报文到监视设备上。

如果没有编写 OB86，当检测到机架故障时 CPU 转为 8TOP 模式。

在相应的块帮助中可以找有关 OB、SFB、SFC 的更详细的信息。

21.9.15 通讯错误（OB87）

说明

当使用通讯功能块或全局数据通讯进行数据交换时出现通讯错误，CPU 的操作系统调用 OB87，例如：

- 当接收全局数据时，检测到不正确的帧标识。
- 用于全局数据状态信息的数据块不存在或太短出。

编程OB87

必须用 STEP 7 在你的 S7 程序中将 OB87 作出一个对象生成。在生成块中编写要在 OB87 中执行的程序，并将它作为你的用户程序的一部分下载到 CPU。

例如，你可以为以下目的使用 OB87：

- 要评估OB87的起动信息。
- 如果找不到用于全局数据通讯状态信息的数据块，要生成该数据块。

如果 OB87 没有编程，当检测到通讯错误时 CPU 转为 STOP 模式。

在相应的块帮助中可以找有关 OB、SFB、SFC 的更详细的信息。

21.9.16 编程错误（OB121）

说明

当出现编程错误时，CPU 的操作系统调用 OB121，例如：

- 寻址的定时器不存在。
- 调用的块未装载。

编程OB121

必须使用 STEP 7 在你的用户程序中将 OB121 作为一个对象生成。在生成的块中编写要在 OB121 中执行的程序，并将它作为你的用户程序的一部分下载到 CPU 中。

例如，你可以为以下目的使用 OB121：

- 要评估OB121的起动信息。
- 要在报文数据块中存入故障原因。

如果不编写 OB121，当检测到编程错误时 CPU 转为 STOP 模式。

在相应的块帮助中可以找有关 OB、SFB、SFC 的更详细的信息。

21.9.17 I/O 访问错误（OB122）

说明

当 STEP 7 指令访问一个输入或输出信号模板，而在最近的一次暖起动中没有这样的模板被赋值，CPU 的操作系统会调用 OB122，例如：

- 直接访问I/O出错（模板损坏或找不到）
- 访问一个CPU不能识别的I/O地址

编程OB122

必须使用 STEP 7 在你的用户程序中将 OB122 作为一个对象生成。在生成的块中编写要在 OB122 中执行的程序,并将它作为你的用户程序的一部分下载到 CPU 中。

你可以为以下目的使用 OB122：

- 要评估OB122的起动信息。
- 要调用系统功能SFC44为输入模板提供一个替代值以便使程序可以用这个有意义的、随过程变化的数值继续执行。

如果不编写 OB122，当检测到 I/O 访问错误时 CPU 转为 STOP 模式。

在相应的块帮助中可以找有关 OB、SFB、SFC 的更详细的信息。

22 打印与归档

22.1 打印项目文献

22.1.1 打印项目文献

当你完成了为你的自动化任务而编制的程序之后，你便可以利用集成于 STEP7 中的打印功能将所有重要数据打印出来，作为项目的文献资料。

你能够打印的项目中的部分

你可以从 SIMATIC 管理器直接打印选定对象的内容，也可以打开相关的对象再起动打印过程。

一个项目中的下列部分能够由 SIMATIC 管理器直接打印出来：

- 选定对象的树形图（项目或库的结构）
- 选定对象的清单（一个对象文件夹的目录）
- 选定对象的内容
- 信息

通过打开相关的对象，一个项目中的下列部分可以打印出来：

- 以梯形图，语句表或功能图表述的程序模块，也能以其它语言表述（选件）
- 带有符号名称和绝对地址的符号表
- 组态表，包括可编程控制器中模块的排列及模块参数
- 诊断缓冲器的内容诊断缓冲区
- 变量表，包括监控格式及监控值和更改值
- 参考数据，例如：交叉表、分配表、程序结构、未使用地址表、无符号地址表
- 全局数据表
- 带有模块状态的模块信息
- 操作者相关文本（用户文本和文本库）
- 选件软件包的文献，例如其它编程语言

DOCPRO选件软件包

为了生成，编辑和打印标准化的接线手册，你可以使用选件软件包 DOCPRO。该软件包可生成符合 DIN 和 ANSI 标准的工厂文献。

22.1.2 打印的基本步骤

打印的步骤如下：

1. 打开相关的对象，将你想要打印的信息显示在屏幕上；
2. 在应用窗口上用菜单命令 File＞Print 打开“打印”对话框。根据你所在的应用窗口不同，菜单条上的第一项可能不是“File”而是要应用的处理对象如“Symbol Table 符号表”。
3. 在打印对话框中，按需要改变打印选项（打印机，打印范围，份数等），然后关闭对话框。

有些对话框有“打印”按钮，例如“模块信息”对话框。点击该按钮即可打印对话框的内容。

程序模块不需要打开。你可以从 SIMATIC 管理将其直接打印出来，使用的菜单命令为 File＞Print。

22.1.3 打印功能

对于打印对象所具有的附加功能见下表：

打印对象	菜单命令	功能	功能	功能	功能
		打印预览	页面设定	页头页脚	打印设定
程序模块 STL 源文件	File > *	•	•	—	•
模板信息		—	•	—	—
全局数据表	GD Table> *	•	•	—	•
组态表	Station > *	•	•	—	•
对象，对象文件夹	File > *	—	•	•	•
参考数据	Reference Data > *	•	•	—	•
符号表	Symbol Table > *	•	•	——	•

打印对象	菜单命令	功能	功能	功能	功能
		打印预览	页面设定	页头页脚	打印设定
变量表	Table> *	---	•	---	•
连接表	网络> *	•	•	---	•
操作者相关文本（用户文本，文本库）	Texts	•	•	---	•
* ： *号是通配符，表示菜单命令中的相关功能（例如，打印预览或页面设定）					

对于每个单独的打印对象的逐步引导说明，可以在“ How to Print（如何打印）”之下找到。

打印预览

利用“ 打印预览 ”功能，可以显示需要打印文献的页面布局。

若文献包含有若干页，则在每页的右下角页码的后边会显示两个圆点记号。最后一页上没有这两个记号，表明无后续页。

注意

文献最终的打印格式在打印预览中不能显示。

设定页面格式

在 SIMATIC 管理器中，使用菜单命令 File > Page Setup，你可以设定所有 STEP 7 应用程序以及你想打印的任选软件包的页面格式（例如 A4、A5、信纸）。在每个应用程序中（例如符号编程器），可以设定只适用于当前段的临时设置。

调整文献的页面布局使其符合打印纸的格式。若文献过宽，右侧的页边会打印到后续页上。

如果你选择带有页边的页面格式（如 A4 带页边），则打印出来的文献在页的左侧留出一个页边，使你可以穿孔和装订。

注意

若需要有关“ 页面设定 ”对话框的帮助，可将光标置于对话框，点击“ 帮助 ”按钮或按 F1 键。

设定页头页脚

利用 SIMATIC 管理器中的“ File＞Headers and Footers ”功能，你可以为想要打印的整个项目文献设定页头页脚。若文献包含有若干页，则在每页的右下角页码的后边会显示两个圆点记号。最后一页上没有这两个记号，表明无后续页。这是一种简捷的方法用以检查打印是否完整。这两个记号在打印预览中也可见。

打印设定

利用“ 打印设定 ”功能，你可以选择打印机和设定纸张格式（水平或直立放置）。这种设定功能，取决于所使用的打印驱动程序。

22.1.4 关于打印选定对象的树形图的特别说明

在“ 打印对象清单 ”对话框中，除了对象清单之外，你还可以通过选中“ 树形图 ”选项而打印出对象的树形图。

若你在“ 打印范围 ”之下选择“ 全部 ”，则整个树形图可被打印出来。若你选择按钮“ 选择 ”，则打印出从被选中对象以下的树形图。

注意

这种在对话框中的设定，仅适用于打印清单和树形图，而不适应于打印对象的内容；打印内容时在相应的应用中有相关的设定。

22.2 项目和库的归档

22.2.1 项目和库的归档

你可以将项目或库以压缩的形式存入一个归档文件中。这种压缩存储过程可以在硬盘上进行也可以在一种便携的数据载体上进行（例如，用软盘）。

归档程序

归档功能为你提供了一个调用你所选定的归档程序的接口。归档程序 ARJ 和 PKZIP 2.50 作为一个组成部分已包含在 STEP 7 软件包内。如果你要使用下列归档程序中的一个，则应选择如下指明的版本（或更新的版本）：

- ARJ 版本2.4.1a以上
- PKZIP 版本2.04g以上
- LHARC 版本2.13以上
- WinZip 版本6.0以上
- JAR 版本1.02以上

关于归档的建议

具有长文件名的项目（文件名字符数多于传统 8.3DOS 格式）或包含有多层目录结构（带有目录其路径名称总长多于 64 字符）的项目，必须使用归档程序 PKZIP2.50，WinZip AR 进行归档。若使用其它归档程序，则不能保证原有结构维持不变，以及归档文件能够正确完整地解压缩。对于包含有从 WinCC 选件软件包得到的对象的项目，这一点应特别注意。

22.2.2 使用保存/归档

另存为（SAVE As）

利用此功能你可以生成一个项目带有另外的名称的副本。

你可以利用此功能：

- 生成备份用副本。
- 复制一个存在的项目以便修改后用于其它目的。

用最快的方法生成一个副本，应选择“ Save as ”选项并且不重新安排对话框。从项目目录向下的整个文件结构都不作任何检查地被复制并存于另外的名称之下。

为了存储备份用副本数据载体上必须有足够的空间。不要试图将项目存于软盘上。因为通常软盘上的空间往往不够。为了将项目数据转移到软盘上应使用“ 归档 ”功能。

重新安排的保存会耗时较长，而且当对象不能被复制和保存时会显示出一条信息。引起的原因可能是一个对象所需的选项软件包或数据不完全。

归档

你可以将项目或库以压缩的形式存入一个归档文件中。这种压缩存储过程可以在硬盘上进行也可以在一种便携的数据载体上进行（例如，用软盘）。

将项目转移到软盘只能用归档文件的形式。若一个项目过大，则在选用归档程序时注意其应具有生成多张软盘上连续归档文件的功能。

被以压缩形式存入一个归档文件的项目或库不能被编辑修改。若你希望对它们进行编辑，就必须将数据解压缩，即恢复项目或库。

22.2.3 对归档的要求

为了将一个项目或一个库进行归档，必须满足以下要求：

- 你必须在系统中安装归档程序。与STEP 7的连接在在线帮助中有说明，题目为“归档/恢复的步骤”。
- 所有与项目有关的数据无一例外地必须在项目目录之内或项目子目录之内。当使用C语言开发环境时，有可能将数据存储在其它位置。这些数将不会被包括到归档文件中。
- 如果你使用归档程序ARJ，PKZip版本2.04g或LHA，文件名称必须符合DOS文件名的规定（8字符文件名加3字符扩展名），因为这些程序是DOS程序。若使用PKZip版本2.50，JAR和WinZip则不必遵守上述对文件名的规定。

22.2.4 归档/恢复的步骤

你可以对你的项目或库进行归档和恢复，使用的菜单命令为File＞Archive或File＞Retrieve。

注意

被以压缩形式存入一个归档文件的项目或库不能被编辑修改。若你希望对它们进行编辑，就必须将数据解压缩，即恢复项目或库。

当进行恢复操作时，被恢复的项目或库会自动被包含到项目/库清单之中。

设定目标子目录

为了设定目标子目录，可在SIMATIC管理器使用菜单命令Options＞Customize。

在这个对话框的“Archive”选项卡之下，你可以接通或关断选项“恢复时检查目录”。

若该选项被关断，则在“项目存储位置”和“库存储位置”同一对话框中所设定的路径将被用于恢复时的目标目录。

将一个归档文件复制到软盘

你可以将一个项目/库进行归档，然后将归档文件复制到软盘上。也可以“ Archive （归档） ”对话框中选择软盘驱动器作为目标目录。

利用PKZIP2.04g的说明

你使用归档程序 PKZIP 在软盘上生成一个归档文件时，若激活了选项 “ 多软盘连续归档 ”，则当你进行恢复时，程序会提示你插入后一张归档的软盘。PKUNZIP 总是在 DOS 窗口内显示下列信息：

“ Insert the LAST disk of the backup set - Press a key when ready. ”（插入后一张备份盘，如果准备好，按任意键）。

当你激活了选项 “ 多软盘连续归档 ”，即便整个归档文件全装在一张软盘上，上述信息，也会显示。

此时，可忽略它，并按任意键而确认对话框。

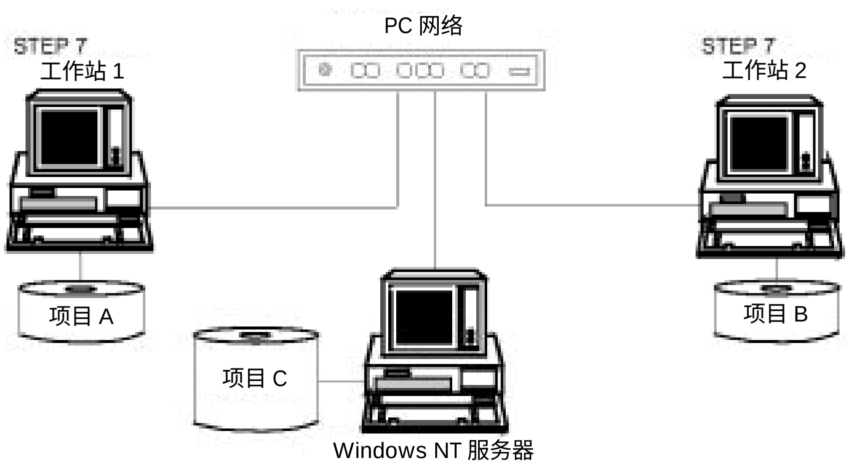
23 多个用户编辑同一个项目

23.1 网络中的多用户配置

概述

借助一个多用户配置方案，你可以在 Windows 95/98/NT/2000 工作组和 NT/Novell 网络中使用 STEP 7 进行工作。这里有三种可能性：

- 项目在一个局部驱动器上并且也要被其它工作站所使用。
- 例如：工作站1和工作站2访问工作站1上的项目A。
  项目位于项目/网络服务器上。
- 例如：工作站1和工作站2访问位于网络服务器上的项目C。
  项目分布于局部驱动器和一个或多个项目/网络服务器之中。
- 例如：工作站1和工作站2访问项目A、B和C。



在网络服务器上存储项目的原则

- 当你将你的项目存到网络服务器上时，访问的路径必须指定为一个驱动器字母。
- 当你将你的项目存到网络服务器上或者存到网络上其它用户的释放的驱动器上时，只有当访问项目的所有STEP 7应用窗口都关闭之后，才能在服务器上或网络结点上关闭Windows95/98/NT/2000 。

多用户编辑S7程序的原则

你需要注意以下事项：

- 在若干用户能够编辑一个S7程序之前，你必须设定工作站的组态（菜单命令Start＞Simatic＞STEP 7＞Configure SIMATIC Workspace）。
- 程序模块和STL源程序文件：
  每个用户应编制不同的程序模块或源文件。若两个用户企图同时编辑同一个程序模块或源程序，将会显示出一条提示信息，且第二个用户的访问被拒绝。
- 符号表：
  若干个用户可以同时打开符号表，但只有一个用户可以对其进行编辑。若两个用户企图同时编辑符号表，则会显示一条提示信息，且第二个用户的访问被拒绝。
- 变量表：
  若干个用户可以同时打开一个变量表，但只有一个用户可以对其编辑。若两个用户企图同时编辑一个变量表，则会显示一条提示信息，且第二个用户的访问被拒绝。在一个S7程序中可以有若干个变量表。当然这些变量表可以被分别进行编辑。

多用户编辑工作站的原则

你需要注意以下事项：

- 一个工作站的硬件组态和网络组态只能集中由一个用户进行编辑。

24　使用 M7 可编程控制系统

24.1　M7 系统程序

具有标准 PC 机结构的 M7-300/M7-400 自动控制微机，在 SIMATIC 自动化平台上，使自由编程功能得到扩展。你可以使用高级语言例如 C 语言或图形化编程语言 CFC（连续功能图）来编制 SIMATIC M7 的用户程序。

为了生成程序，除 STEP 7 之外，你还需要用于 M7-300/400 的系统软件 M7-SYSRT 和用于 M7 程序的开发环境（Pro C/C++或 CFC）。

基本程序

当你要利用 SIMATIC M7 生成一个自动控制方案时，有一系列基本任务要完成。下表指出了对于大多数项目都要执行的任务，将其定义为基本步骤。下表还给出了在本手册或其它手册中可供参考的相关章节。

步　骤	说　明
设计自动控制方案	M7 专用； 参考： M7-SYS RT 编程手册
启动 STEP 7	同 S7
创建项目结构 设置站 建立硬件组态	同 S7
建立通讯连接组态	同 S7
定义符号表	同 S7
生成 C 语言或 CFC 用户程序	M7 专用； 参考：ProC/C++
配置操作系统 在 M7-300/M7-400 中安装操作系统 向 M7 下装硬件组态和用户程序	M7 专用； 参考： M7-SYS RT 用户手册
检测并调试用户程序	ProC/C++
运行监控和 M7 诊断	同 S7，但是没有用户自定义的诊断
打印与归档	同 S7

在M7中有什么不同？

对于 M7-300/M7-400 STEP 7 不支持如下功能：

- 多重运算—若干个CPU的同步操作
- 变量强置
- 全局数据通讯
- 用户自定义的诊断

M7可编程控制系统的管理

STEP 7 为使用 M7 可编程控制系统提供如下特别的支持：

- 在M7-300/M7-400中安装操作系统
- 通过编辑系统文件对操作系统进行配置
- 向M7-300/M7-400装载用户程序
- 软件升级

为了访问 M7 可编程控制系统的管理器，在包含有 M7 CPU 或 FM 模块工作站的项目中，选中 M7 程序文件夹，选择菜单命令：PLC＞Manage M7 System 在有关 M7-SYSRT 的用户手册和联机帮助中有详细的说明。

24.2 用于 M7 编程的选装软件

M7选装软件

STEP 7 为你提供了所需的如下基本功能：

- 项目的生成与管理
- 对硬件进行配置和参数设定
- 配置网络和连接
- 符号数据管理

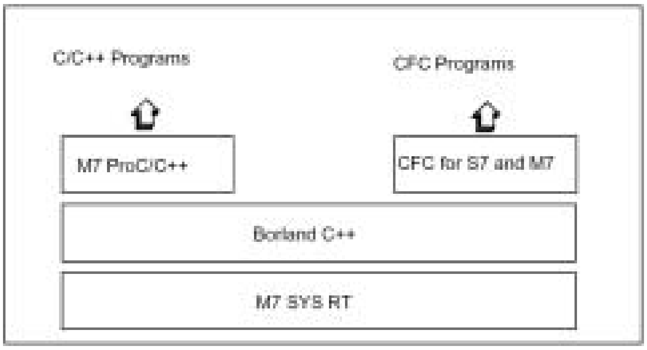
不论是使用 SIMATIC S7 还是 SIMATIC M7 这些功能都能具备为了实现 M7 有关的应用。除了 STEP 7 之外，你还需要 M7 选装软件。

软件	内　容
M7-SYS RT	· M7 RMOS32 操作系统 · M7-API 系统库 · 对 MPI 的支持
用于 S7 和 M7 的 CFC	CFC（连续功能图）编程软件
M7-ProC/C++	· 用于在 STEP 7 中实现 Borland 开发环境的连接 · 符号的编辑与生成 · xdb386 高级语言调试工具的推理规则
Borland C++	Borland C/C++开发环境

与 M7 选装软件配合，STEP 7 还可以支持下列附加功能：

- 通过多点接口（MPI）向M7可编程控制系统下装数据
- 查询有关M7可编程控制系统的信息
- 在M7可编程控制系统上进行特别设定和对M7进行复位

下图显示了 M7 编程对 M7 选装软件的依赖性。



小结

为生成…	所需的 M7 选装软件…
C/C++程序	1. M7-SYS RT 2. M7-ProC/C++ 3. Borland C++
CFC 程序	1. M7-SYS RT 2. 用于 S7 和 M7 的 CFC 3. Borland C++

何种软件提供何种支持

为 M7 应用所持需的工具，一部集成于 STEP 7 之中，另一部分是 M7 的选装软件。

下表指明了哪个软件包支持哪种功能：

软件	提供的支持
STEP 7	· 安装 M7 操作系统 · 管理 M7 可编程控制系统 · 下装，起动和删除 M7 程序 · 显示状态和诊断数据 · CPU 复位
M7-SYS RT	M7 操作系统和 M7 系统软件应用帮助实现以下功能： · 程序处理的控制 · 内存和资源的管理 · 对计算机和 SIMATIC 硬件的访问 · 处理中断 · 诊断 · 状态监控 · 通讯
M7-ProC/C++	· 集成代码的生成(将 Borland 的开发环境集成于 STEP 7 中) · 将项目符号连接到源代码 · 集成的调试功能
Borland C++	· 生成 C 和 C++程序
用于 S7 和 M7 的 CFC	· 生成，检测和调试 CFC 程序 · 起动并运行 CFC 程序

24.3 M7-300/M7-400 操作系统

对于用高级语言 C 和 C++生成的应用，操作系统的作用是非常重要的。对于这些应用，操作系统要承担下列任务：

- 访问硬件
- 管理资源
- 系统集成
- 与系统中其它部件进行通讯

为了完成自动控制任务，M7 RMOS32（实时多任务操作系统）实时操作系统与 SIMATIC M7 自动控制微机配合应用。M7 RMOS32 经过扩展后包含了一个调用接口，M7 API（应用程序接口）将其集成到 SIMATIC 系统之中。

实时操作系统 M7 RMOS32 是用于实时和多任务方案的，在时间准则上使用 32 位处理。对于 M7 模块它可以有如下列配置：

- M7 RMOS32
- M7 RMOS32带MS-DOS

你选择的 M7 可编程控制系统的操作系统配置取决于你所用的 M7 模块：

操作系统配置	模块/主内存	PROFIBUS-DP 和 TCP/IP 有/无	大容量内存的安装
M7 RMOS32	FM 356-4 / 4 MB FM 356-4 / 8 MB CPU 388-4 / 8 MB FM 456-4 / 16 MB CPU 488-3 / 16 MB CPU 486-3 / 16 MB	无 有 有 有 有 有	内存卡≥4MB 或在硬盘上
M7 RMOS32 带 MS-DOS	FM 356-4 / 8 MB CPU 388-4 / 8 MB FM 456-4 / 16 MB CPU 488-3 / 16 MB CPU 486-3 / 16 MB	无 无 有 有 有	内存卡≥4MB 或在硬盘上

25 提示与技巧

25.1 更换组态表中的模板

如果你使用 HW Config 修正一个站的组态，并且你想更换一个模板，应如下进行：

1. 使用拖放功能，将模板从 Hardware Catalog（硬件目录）窗口中拖到旧模板上。
2. 放下新模板。应尽可能地使新模板使用已插模板的参数。

删除旧模板，插入新模板并赋值参数，比更换模板速度快。

你可以使用菜单命令 Options > Settings（“Enable Module Swapping（使能模板交换）”），你可以在 HW Config 中关闭或打开该功能。

25.2 具有大量网络站的项目

如果你逐一组态所有站，并使用菜单命令 Options > Configure Network 调用 NetPro，以便组态连接，站将自动放置在网络视图中。该程序的缺点是你必须根据拓扑条件安排站和子网。

如果你的项目中包括大量的网络站，并且你想在这些站之间组态连接，你应在网络视图中组态系统结构，并且开始保存概览。

1. 在 SIMATIC 管理器中生成新项目（菜单命令 File＞New）。
2. 启动 NetPro（菜单命令 Options > Configure Network）
3. 如下在 NetPro 中一个站一个站地生成：

   - 使用拖放功能，从 Catalog 窗口中拖出站。
   - 双击站，启动 HW Config。
   - 使用拖放功能，在 HW Config 中放置具有通讯功能的模块（CPU、CP、FM、IF 模块）。
   - 如果你想网络连接这些模板，双击组态表中的相应行，生成新的子网，并网络连接接口。
   - 保存组态，并切换到 NetPro。

- 在 NetPro 中，定位站和子网（使用鼠标移动对象，直至到达所需位置）

4. 在 NetPro 中组态连接，根据需要校正网络连接。

25.3 再排列

若在使用 STEP 7 过程中，出现一些无法解释的问题，对项目或库的数据进行重新安排，往往会对克这些问题有利。

选择菜单命令 File＞Rearrange 可进行重新安排。该功能可消除由于某些内容被删除后而产生的数据存储不连续，即使得项目/库所需的存储器量减少。

这个功能对项目或者库的数据存储进行优化，其方法类似硬盘碎片整理程序对硬盘的文件存储进行优化。

重新安排处理持续的长短取决于要移动的数据量，很可能要花些时间。因此该功能不能自动实施（例如，当你关闭一个项目时），而应由用户在打算重新安排项目或库时进行调用。

要求

项目和库需要进行重新安排，必须保证其中不能有正在被其它用户编辑的对象，因为这时的存取权会被封锁。

25.4 用变量表进行测试

监视和修改变量表中的变量时，请注意以下编辑技巧：

- 可以在“符号”栏输入符号和地址，也可以在“地址”栏输入符号和地址。然后输入便会自动地在正确的栏中写入。
- 想显示修改值，你应将“Monitoring（监视）”触发点设在“Beginning of Scan Cycle（扫描循环开始）”，将“Modifying（修改）”触发点设在“End of Scan Cycle（扫描循环结束）”。
- 如果你把光标放在红色的行上，可以从工具技巧中读到错误的原因。按动F1键，可获得消除这些错误的建议。
- 你只能输入已在符号表中定义过的符号。符号必须准确地按照它在符号表中所定义的行输入。有特殊字符的符号名必须用引号括起来（例如，“Motor.Off”，“Motor+Off”，“Motor-Off”）。

- 在“ Online（在线） ”选项卡（“ Customize（自定义） ”对话框）中可以关闭警告。
- 使用事先断开连接，就可改变连接。
- 监控变量时，可以定义监控触发器。
- 通过选择行，并执行“ Force（强制） ”功能，可以修改所选变量。只有高亮显示的变量才能修改。
- 使用“ Combine Variables（组合变量） ”选项，你可以增加可以监视的变量数量（“ Customize（自定义） ”对话框中的“ Online（在线） ”选项卡。
- 无确认退出：
  在“ Monitoring（监视） ”、“ Modifying（修改） ”、“ Release PQ（释放 PQ） ”时，如果你按动 ESC 键，将不会询问你是否想退出而中止“ Monitoring（监视） ”和“ Modifying（修改） ”。
- 输入一个“ Contiguous Address Range（连续的地址范围） ”：
  使用菜单命令 Insert>Range of Variable。
- 显示和隐藏栏：
  使用以下菜单命令可以显示和隐藏单个栏：

符号：	View > Symbol
符号注释：	View > Symbol Comment
状态数值的输出格式：	View > Display Format
变量的状态值：	View > Status Value
修改变量值：	View > Modify Value

- 同时修改表中几行的显示格式：
  - 按住鼠标左键在所需表中拖动，选中那些要改变显示格式的区域。
  - 使用菜单命令 View > Select Display Format，选择输出格式。只有那些表中允许改变格式的行才能改变格式。
- 按动F1键，可以获得输入示例：
  - 如果将光标放在地址栏，并且按动 F1 键，你可以获得地址输入的示例。
  - 如果将光标放在修改值栏，并且按动 F1 键，你可以获得修改/强制值输入的示例。

25.5 虚拟工作存储器

引起 STEP 7 出问题的另一个原因有可能是虚拟工作存储器过小。

为了在 Window 95/98/NT/2000 之下使用 STEP 7，你应该调整虚拟存储器的设定。调整的步骤如下：

1. 打开“控制面板”，利用命令 Start > Settings > Control Panel。
2. 双击“System（系统）”图标。
   在 Windows95/98/NT 下，选择“System Properties（系统性能）”对话框中的“Performance（性能）”选项卡。
3. 在“Performance Characteristics（性能特性）”
   在 Windows2000 下，选择“Advanced”选项卡，并点击“System Performance Options（系统性能选项）”按钮。
4. 点击“Virtual Memory（虚拟存储器）”（Windows 9x 下）按钮或“Change（修改）”（Windows NT/2000 下）按钮。
5. 只适用于 Windows 9x：在“虚拟存储器”对话框中，选中选项“用户自定义虚拟存储器”
6. 在“最小”处输入至少 40MB，在“最大”处至少 150MB
7. 只适用于 Windows 9x：确保选项“禁止使用虚拟存储器”不能生效只适于 NT：点击“Set（设置）”按钮。

注意

对于在硬盘（默认为 C：）上且为动态的虚拟存储器，你必须确保为子目录 TMP 或 TEMP 提供足够的存储器量（大约 20 至 30MB）

- S7项目也在虚拟存储器设定的同一分区内，则应保证有大约两倍于S7项目大小的存储器空间。
- S7项目存于其它分区，此项要求无关。
